Systême soft

Программный комплекс Систэм Платформ

SePlatform.Data Server 2.1

Руководство администратора

Редакция	Соответствует версии ПО
3. Предварительная	2.1.2

Содержание

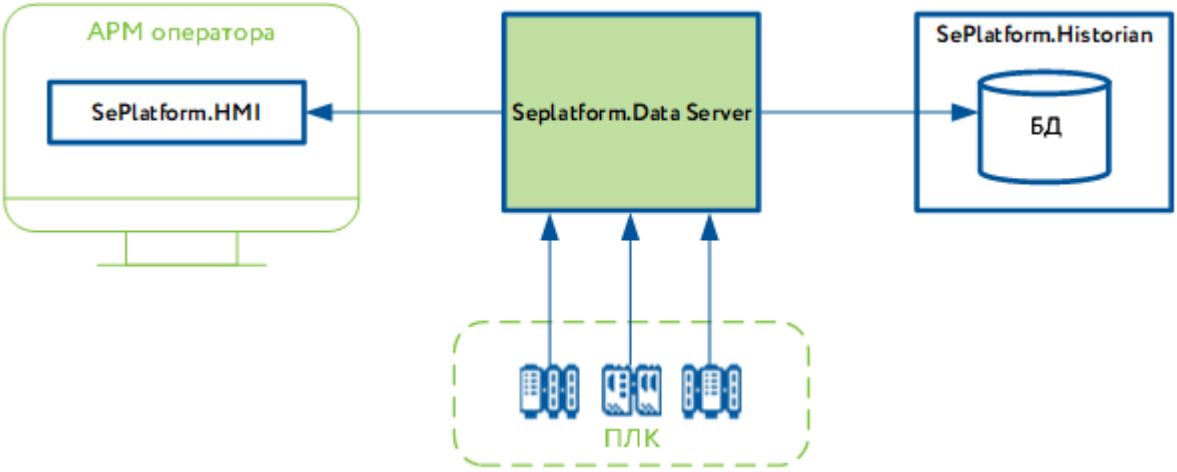
1. Об SePlatform.Data Server

SePlatform.Data Server – это программный комплекс для обмена данными между технологическим оборудованием и компонентами проекта. С его помощью можно:

- собирать данные о состоянии технологического процесса;
- предоставлять собранные данные персоналу;
- доставлять команды персонала оборудованию.

2. SePlatform.Data Server в составе Систэм Платформ

С помощью SePlatform.Data Server компоненты Систэм Платформ взаимодействуют друг с другом и с технологическим оборудованием. Например, получив данные от ПЛК, SePlatform.Data Server отправляет их в SePlatform.HMI для предоставления персоналу и в SePlatform.Historian для архивации.



2.1. Назначение и преимущества

Основной задачей SePlatform.Data Server, как компонента Систэм Платформ, является выполнение функций сбора, обработки и предоставления данных. SePlatform.Data Server выполняет сбор технологических данных с коммуникационных устройств в ходе мониторинга контролируемых объектов. На основе полученной информации осуществляется контроль над технологическим процессом. Управление может происходить как по команде оператора, при передаче собранных данных на верхние уровни АСУ ТП, так и по встроенным алгоритмам.

SePlatform.Data Server является шлюзом для работы SCADA-системы с устройствами ввода/вывода информации. Одновременно сервер может поддерживать соединение с несколькими промышленными контроллерами. Установка нескольких экземпляров SePlatform.Data Server на одном компьютере решает задачу конвертации протоколов (например, Modbus в ГОСТ Р МЭК или в OPC).

SePlatform.Data Server используется как часть автоматизированной системы управления, построенной на базе Систэм Платформ. В основе АСУ данного уровня лежит максимальная интеграция компонентов системы, вследствие этого согласованность в обработке данных и выполнении команд. Компонент SePlatform.Data Server, благодаря данному принципу, характеризуется высокой производительностью при обмене данных с объектами управления, расположенными на верхних уровнях системы управления. В процессе проектирования Систэм Платформ была учтена угроза потери данных в случае сбоев в работе. В результате при внедрении АСУ в среде исполнения применяется резервирование сервера и каналов связи. Посредством резервирования обеспечивается также высокий уровень надёжности и гарантируется своевременная и бесперебойная доставка событий и команд.

SePlatform.Data Server может использоваться как в локальных системах управления, так и в распределённых промышленных предприятиях. SePlatform.Data Server, входящие в состав АСУ, работающих на различных объектах единого предприятия, поддерживают соединение друг с другом на сетевом уровне.

2.2. Принцип работы

Полноценное функционирование SePlatform.Data Server не ограничено временными границами. Сервер может стабильно работать в режиме 24/7 (двадцать четыре часа, семь дней в неделю) с сохранением всех возможностей. Перезапуск сервера требуется только для выполнения конфигурационных настроек (стр. 22).

SePlatform.Data Server характеризуется высокой производительностью, о чём свидетельствует возможность ядра выполнять более 2 000 000 операций в секунду. Корректная работа сервера не нарушается при обработке значений более 1 000 000 сигналов и выполнении ядром более 1 000 000 операций уведомления. Управляющие команды, получаемые сервером, с учётом выделенного им места в очереди отправляющим объектом, оперативно передаются на уровень ниже. При передаче команд управления гарантируется высокая скорость передачи и сохранность передаваемых данных.

Работа с SePlatform.Data Server организована в двух режимах: **основной** и **резервный**. Благодаря резервированию серверов повышается их надёжность и обеспечивается сохранность собранных технологических данных в случае сбоев в работе. Включение в работу резервного сервера производится диспетчером вручную, либо автоматически в случае отсутствия связи с одним из серверов. Работая в режиме резерва, сервер задаёт условия работы коммуникационных модулей, входящих в его состав. Модули, находящиеся в резервном режиме, не отправляют управляющих воздействий. В результате, механизм резервирования также снижает нагрузку на оборудование, что увеличивает скорость выполнения операций.

Основными составляющими SePlatform.Data Server являются программные модули. Один экземпляр SePlatform.Data Server содержит до 64 модулей. Набор модулей не является постоянным и подбирается в соответствии с задачами проекта автоматизации (стр. 24).

В комплект поставки SePlatform.Data Server входят также сервисные приложения для управления, конфигурирования, просмотра статистической информации. Данные приложения предназначены для установки на автоматизированных рабочих местах администраторов. Все сообщения о работе сервера и его компонентов фиксируются в журнал приложений ОС.

3. Архитектура SePlatform.Data Server

SePlatform.Data Server - компонент Систэм Платформ, выполняющий следующие задачи:

- сбор данных с устройств в ходе мониторинга контролируемых объектов;
- предоставление данных клиентам по различным протоколам и спецификациям;
- повышение надежности проекта за счёт резервирования;
- логическая обработка данных в режиме реального времени;
- генерация событий и тревог на основе полученных данных.

Сервер построен по модульному принципу, что позволяет конфигурировать его в зависимости от выполняемых задач и не создавать лишней нагрузки.

Количество экземпляров сервера на одном компьютере не ограничено, что позволяет использовать сервер в качестве конвертера протоколов или для создания демилитаризованных зон.

Сбор данных

SePlatform.Data Server обеспечивает опрос источников данных по различным протоколам и спецификациям.

Протокол/спецификация	**Модуль сервера**	**Поддержка в ОС**	
		Windows	**Linux**
ГОСТ Р МЭК 60870-5-101	IEC-101 Master	✓	✓
ГОСТ Р МЭК 60870-5-104	IEC-104 Master	✓	✓
ГОСТ Р МЭК 61850	IEC-61850 Client	✓	✓
Modbus TCP	Modbus TCP Master	✓	✓
Modbus RTU	Modbus RTU Master	✓	✓
OPC DA	OPC DA Client	✓	
OPC HDA	OPC HDA Client	✓	
OPC UA	OPC UA Client	✓	✓
SQL	SQL Connector	✓	✓
SNMP	SNMP Manager	✓	✓
Syslog	Syslog Server	✓	✓
BACnet	BACnet Client	✓	✓
FINS	FINS Client	✓	✓
NFL	NFL Client	✓	✓

Протокол/спецификация	Модуль сервера	Поддержка в ОС	
		Windows	Linux
S7	Siemens S7 Client	✓	✓
UNET	UNET Client	✓	✓
EtherNet/IP	EtherNet/IP Scanner	✓	✓

Предоставление данных клиентам

SePlatform.Data Server предоставляет данные клиентам по различным протоколам и спецификациям.

Протокол/спецификация	Модуль сервера	Поддержка в ОС	
		Windows	Linux
ГОСТ Р МЭК 60870-5-101	IEC-101 Slave	✓	✓
ГОСТ Р МЭК 60870-5-104	IEC Slave	✓	✓
Modbus TCP	Modbus TCP Slave	✓	✓
Modbus RTU	Modbus RTU Slave	✓	✓
OPC DA	OPC DA Server	✓	
OPC HDA	HDA Server	✓	
OPC AE	OPC AE Server	✓	
OPC UA	OPC UA	✓	✓
TCP	TCP Server	✓	✓
Файловый интерфейс	TCP Server	✓	✓

Ядро

Ядро SePlatform.Data Server является центральным компонентом сервера. Предназначено для реализации инфраструктуры сервера, интерфейсов работы с модулями, сигналами и их свойствами, остальными подсистемами. Ядро может производить значимые логические вычисления, требующие наибольшей скорости вычислений. Такой подход позволяет значительно повысить производительность работы сервера. Все вычисления производятся по описанным при конфигурировании алгоритмам.

Основные функции ядра SePlatform.Data Server:

- пересчет значений из физических значений в инженерные и в обратном направлении. При пересчете используются линейная и линейная с изломом зависимости;
- выполнение алгоритмов по событию, таймеру и расписаниям;
- управление запуском и остановом модулей при старте и в процессе работы сервера;
- управление состоянием сервера в рамках резервирования;

- запись и чтение данных из ОБД;
- управление модулями, отправка и принятие уведомлений об изменении значений сигналов.

Резервирование

Модуль сервера	Поддержка в ОС	
	Windows	Linux
Модуль резервирования	✓	✓

SePlatform.Data Server реализует два вида резервирования:

- горячее резервирование;
- полное дублирование.

При горячем резервировании система позволяет настроить репликацию данных между резервируемыми серверами для поддержания оперативной базы данных резервного сервера в актуальном состоянии. Тонкая настройка сервера позволяет ограничивать функции сервера в состоянии резерва в широком диапазоне (полное или частичное отключение опроса и обработки данных).

При полном дублировании, серверы работают независимо друг от друга и оба доступны для работы с клиентами. В этом случае клиентское приложение само решает с каким сервером работать. При реализации крупных распределенных проектов с организацией резервируемых пунктов управления возможно создание единой системы резервных пар серверов.

Логическая обработка данных

Модуль сервера	Поддержка в ОС	
	Windows	Linux
Logics Module	✓	✓
Data Buffer	✓	✓
Write VQT	✓	✓

Одна из первостепенных задач SePlatform.Data Server - промежуточная обработка данных. Для повышения производительности работы SePlatform.Data Server все вычисления, производимые при обработке параметров, вынесены на уровень ядра. За внутресерверные вычисления отвечает Logics Module. Алгоритмы модуля логики составляются на специальном скриптовом языке SePlatform.Om.

Возможности логической обработки данных:

- пересчет значений из физических в инженерные и обратно (по линейной и линейной с изломом зависимостям);
- пересчет значений сигналов по формуле;
- выполнение алгоритмов по событию, таймеру или расписанию;
- вызов функций из внешних динамических библиотек;
- перехват генерируемых событий и тревог.

Специфичные задачи логической обработки:

- разбор буфера для выделения кода технологического объекта и кода события (тревоги) (модуль Data Buffer);
- опциональное изменение свойств сигнала **`Value`**, **`Quality`** или **`Timestamp`** (модуль Write VQT).

Генерация событий и тревог

Модуль сервера	Поддержка в ОС	
	Windows	Linux
OPC AE Server	✓	✓
Модуль рассылки событий	✓	✓

ОБРАТИТЕ ВНИМАНИЕ

В ОС Linux модуль OPC AE Server генерирует события, но не предоставляет их.

На основе полученных и обработанных данных, сервер может по заранее определенным правилам и алгоритмам генерировать и предоставлять пользователям сообщения о событиях и тревогах. Сервер генерирует события по нескольким алгоритмам срабатывания: дискретный переключатель, перечисление, отклонение, по уровню (модуль OPC AE Server).

Возможности сервера по генерации событий и тревог:

- генерация событий в рамках спецификации OPC AE;
- предоставление информации о событиях в рамках спецификации OPC DA (только в ОС Windows);
- отправка информации о событиях по электронной почте (Модуль рассылки событий).

Прочие возможности SePlatform.Data Server

Модуль сервера	Поддержка в ОС	
	Windows	Linux
SnapShot	✓	✓
FS Generator	✓	
NetDiag	✓	
NetDiag2	✓	✓
Модуль истории	✓	✓
ProcessMonitor	✓	✓

Прочие возможности SePlatform.Data Server:

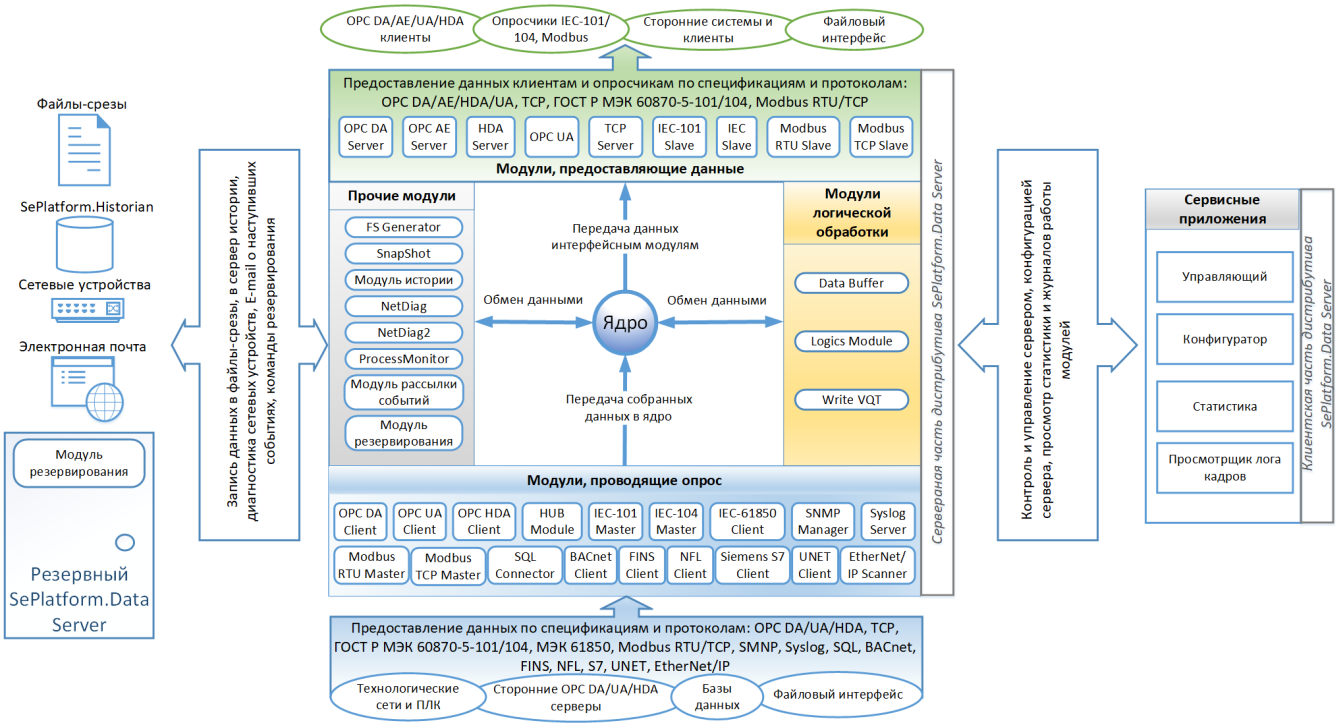
- сохранение текущих значений сигналов в файл-срезы XML-формата (модуль SnapShot);
- сохранение текущих значений сигналов в файл-срезы бинарного формата (модуль FS Generator);
- диагностика сетевых устройств (модули NetDiag, NetDiag2);
- предоставление данных для записи в сервер истории (Модуль истории).

Сервисное обслуживание SePlatform.Data Server

Обслуживание сервера выполняется сервисными приложениями, которые входят в состав клиентской части дистрибутива SePlatform.Data Server (только в ОС Windows):

- Редактирование конфигурации сервера выполняется с помощью сервисного приложения Конфигуратор.
- Просмотр статистической информации о работе сервера выполняется с помощью сервисного приложения Статистика.
- Просмотр журналов работы модулей сервера выполняется с помощью сервисного приложения Просмотрщик лога кадров.
- Комплексное обслуживание, администрирование и управление сервером или резервной парой серверов выполняется с помощью сервисного приложения Управляющий.
- Также для сервисных и диагностических целей при работе с проектами автоматизации применяется набор инструментов SePlatform.Tools.

3.1. Архитектурная схема сервера



4. Подготовка к работе

4.1. Системные требования

SePlatform.Data Server функционирует в виде:

- службы **`SePlatform.Server`** в ОС Windows;
- сервиса **`seplatform.server.service`** в ОС Linux.

На одном компьютере возможно функционирование нескольких экземпляров SePlatform.Data Server (стр. 15). Количество серверов на одной машине ограничено её производительностью. Совместная нагрузка процессора при наличии нескольких экземпляров сервера не должна превышать 70%. Работа SePlatform.Data Server возможна без открытия сеанса пользователя.

Системные требования компьютеров для установки серверной части SePlatform.Data Server:

ОС	Microsoft Windows 10 Pro/11 Pro Microsoft Windows Server 2012/2012 R2/2016/2019/2022 Astra Linux, РЕД ОС, Ubuntu, ОС семейства "Альт" (glibc не ниже 2.17)
Разрядность ОС	x64
Процессор	Intel Celeron с тактовой частотой не менее 1.6 ГГц
Объем оперативной памяти	не менее 2 ГБ
Объем дисковой памяти	не менее 1 ГБ
Сетевой адаптер	Ethernet 10/100/1000 Мбит/с
Установленное ПО	Для ОС Windows: › Антивирусное ПО › OPC Core Components версии 105.1 (ссылка для скачивания: https://opcfoundation.org/developer-tools/samples-and-tools-classic/core-components)

Системные требования компьютеров для установки клиентской части SePlatform.Data Server:

ОС	Microsoft Windows 10 Pro/11 Pro Microsoft Windows Server 2012/2012 R2/2016/2019/2022
Разрядность ОС	x64
Процессор	Intel Celeron с тактовой частотой не менее 1.6 ГГц

Объем оперативной памяти	не менее 1 ГБ
Объем дисковой памяти	не менее 500 МБ
Сетевой адаптер	Ethernet 10/100/1000 Мбит/с
Установленное ПО	Антивирусное ПО .NET 4.6.1 (ссылка для скачивания: https://www.microsoft.com/ru-ru/download/details.aspx?id=49982) OPC .NET API 2.00 Redistributables 105.0 (ссылка для скачивания: https://opcfoundation.org/developer-tools/samples-and-tools-classic/net-api-sample-client-source-code)

4.2. Стандартная установка

ОС Windows

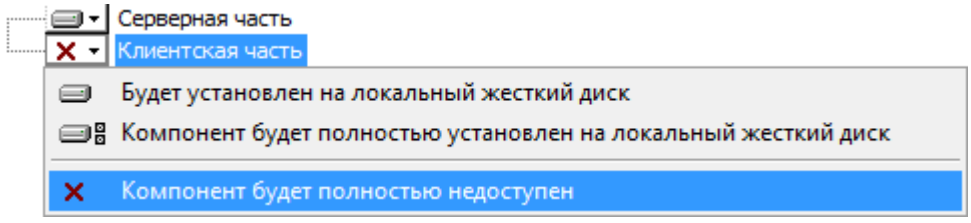
ОБРАТИТЕ ВНИМАНИЕ

Для установки SePlatform.Data Server следует выполнить вход в систему с правами администратора ОС.

Для установки SePlatform.Data Server:

1. Запустите дистрибутив `SePlatform.Server-x.x.x+xx.xxxxx (x64).msi`.
2. По умолчанию устанавливаются все компоненты SePlatform.Data Server:
   - серверная часть;
   - клиентская часть, состоящая из сервисных приложений:
     - Управляющий;
     - Конфигуратор;
     - Просмотрщик лога кадров;
     - Статистика.

   Если необходимо установить только один из компонентов, в параметрах установки заблокируйте компонент, который устанавливать не нужно:



3. Следуйте указаниям мастера установки.

Каталоги установки компонентов SePlatform.Data Server по умолчанию:

- серверная часть - `C:\Program Files\SePlatform\SePlatform.Server\Server`;
- клиентская часть - `C:\Program Files\SePlatform\SePlatform.Server\Service`.

ОС Linux

В ОС Linux устанавливается только серверная часть SePlatform.Data Server. Установка выполняется штатным пакетным менеджером.

> ОБРАТИТЕ ВНИМАНИЕ
> Команда установки выполняется только от суперпользователя **«root»**.

Имя устанавливаемого пакета: `seplatform.server-x.x.x+xx.xxxxx.deb` или `seplatform.server-x.x.x+xx.xxxxx.rpm` в зависимости от используемой ОС Linux. Находясь в папке с установочным пакетом, запустите установку.

Установка пакета `*.rpm` с помощью пакетного менеджера YUM:

```
yum install seplatform.server-x.x.x+xx.xxxxx.rpm
```

Установка пакета `*.rpm` с помощью пакетного менеджера RPM:

```
rpm -i seplatform.server-x.x.x+xx.xxxxx.rpm
```

Установка пакета `*.deb` с помощью пакетного менеджера apt:

```
apt-get install seplatform.server-x.x.x+xx.xxxxx.deb
```

Установка пакета `*.deb` с помощью пакетного менеджера dpkg:

```
sudo dpkg -i seplatform.server-x.x.x+xx.xxxxx.deb
```

SePlatform.Data Server устанавливается в директорию `/opt/SePlatform/SePlatform.Server`.

4.3. Установка дополнительной копии SePlatform.Data Server

ОС Windows

Чтобы установить два экземпляра SePlatform.Data Server на одном компьютере под управлением ОС Windows, выполните действия:

1. Установите SePlatform.Data Server стандартным способом. В данном описании предполагается, что SePlatform.Data Server будет установлен в папку по умолчанию `C:\Program Files\SePlatform\SePlatform.Server`.
2. Скопируйте папку `C:\Program Files\SePlatform\SePlatform.Server` со всем содержимым в новую папку, например, с именем `C:\Program Files\SePlatform\SePlatform.Server1`.

3. Откройте настроечный xml-файл копии сервера: `C:\Program Files\SePlatform\SePlatform.Server1\Server\SePlatform.Server.xml`.
4. Внесите следующие изменения в настроечный файл:
   - Измените значение атрибута **ServiceName** (атрибут элемента **install**) - имя службы сервера. Например:

```
ServiceName = "SePlatform.Server1"
```

   - Измените значения идентификатора **ProgID** и класса **CLSID** для каждого регистрируемого COM-сервера. Например, для COM-сервера OPCDA:

```
ProgID="SePlatform.OPCDAServer.1" CLSID="{ED738D76-9CDF-4C05-92EB-5268345B6F4F}";
```

   для COM-сервера OPCAE:

```
ProgID="SePlatform.OPCAEServer.1" CLSID="{A439CED6-0D7F-43A5-BD97-BEA80507C04A}"
```

   для COM-сервера HDA:

```
ProgID="SePlatform.HDAServer.1" CLSID="{95774D45-B3FF-4B82-9B44-09B07DAF9C58}"
```

   - Измените значение атрибута **Port** (атрибут элемента **Connection**) - номер порта подключения к серверу. Например:

```
Connection Port="4573"
```

Вид файла `SePlatform.Server.xml` копии сервера с измененными атрибутами имеет вид:

```
<?xml version="1.0" encoding="windows-1251"?>
<configuration>
  <install ServiceName="SePlatform.Server1" ExeName="SePlatform.Server.exe">
    <ComServers>
      <OPCDA ProgID="SePlatform.OPCDAServer.1" CLSID="{28A2AD9C-C45E-4C6b-
A0C3-6E363F99CA72}" />
      <OPCAE ProgID="SePlatform.OPCAEServer.1" CLSID="{0CAEA48A-D7E6-44A4-
85FD-C27836727D07}" />
      <HDA ProgID="SePlatform.HDAServer.1" CLSID="{8002740A-886F-1488-2280-
42058D6D5CA8}" />
    </ComServers>
  </install>
  <Storage Filename="SePlatformServer.cfg" />
  <Connection Port="4573" />
  <Backup Path="..\Backups" Time="00:00" StorageDepth="14" />
  <Log Path="..\Logs" />
  <Dispatch Model="Default" />
  <Config ReadOnly="False" />
  <Instance ID="589745A2-5A75-4C4E-8540-B3795A0B2EF1" />
  <Culture LangID="ru-RU" />
</configuration>
```

5. Зарегистрируйте копию сервера. В командной строке, находясь в папке `C:\Program Files\SePlatform\SePlatform.Server1\Server`, выполните команду:

```
SePlatform.ServerInstaller.exe /install
```

6. Запустите копию сервера.
7. Подключитесь тестовым клиентом и убедитесь в доступности модулей и сигналов сервера.

ОС Linux

Чтобы добавить дополнительный экземпляр сервиса SePlatform.Data Server на одном компьютере под управлением ОС Linux, выполните следующие действия:

1. Скопируйте существующую папку сервера командой:

```
cp -R /opt/SePlatform/SePlatform.Server /opt/SePlatform/SePlatform.Server2
```

2. Перейдите в добавленную папку и измените в файле `SePlatform.Server.xml`:
   - имя сервиса в атрибуте **ServiceName** (атрибут элемента **install**):

```
ServiceName = "SePlatform.Server2"
```

   - номер порта подключения к серверу в атрибуте **Port** (атрибут элемента Connection):

```
Connection Port="4573"
```

3. Скопируйте unit-файл сервиса SePlatform.Data Server командой:

```
sudo cp /lib/systemd/system/seplatform.server.service
/lib/systemd/system/seplatform.server2.service
```

4. Откройте добавленный файл командой:

```
sudo nano /lib/systemd/system/seplatform.server2.service
```

В открывшемся файле измените:

- описание сервиса в параметре **Description** (раздел **Unit**)

```
Description=SePlatform.Server2
```

- путь к папке сервера в параметре **WorkingDirectory** (раздел **Service**)

```
WorkingDirectory=/opt/SePlatform/SePlatform.Server2
```

- путь к библиотекам сервера в параметре **LD_LIBRARY_PATH** (раздел **Service**)

```
Environment=LD_LIBRARY_PATH=/opt/SePlatform/SePlatform.Server2
```

- путь к исполняемому файлу в параметре **ExecStart** (раздел **Service**)

```
ExecStart=/opt/SePlatform/SePlatform.Server2/SePlatform.Server
```

5. Зарегистрируйте и запустите новый экземпляр сервиса SePlatform.Data Server командами:

```
sudo systemctl enable seplatform.server2
```

```
sudo systemctl start seplatform.server2
```

4.4. Запуск и останов SePlatform.Data Server

ОС Windows

Управление сервером выполняется путем запуска/перезапуска/останова службы **SePlatform.Server** стандартными инструментами ОС Windows.

Управление сервером в составе резервной пары возможно через сервисное приложение Управляющий.

ОС Linux

Управление сервером выполняется путем запуска/перезапуска/останова сервиса seplatform.server специализированными командами.

ОБРАТИТЕ ВНИМАНИЕ

Все команды выполняются только от суперпользователя **«root»**.

Запуск:

```
systemctl start seplatform.server
```

Останов:

```
systemctl stop seplatform.server
```

Перезапуск:

```
systemctl restart seplatform.server
```

4.5. Удаление SePlatform.Data Server

ОС Windows

Удаление SePlatform.Data Server и вспомогательного ПО выполняется стандартными инструментами:

1. Запустить программу Программы и компоненты: `Пуск` → `Панель управления` → `Программы и компоненты`.
2. Из представленного списка установленных программ выбрать SePlatform.Data Server и нажать кнопку **Удалить**.

При удалении SePlatform.Data Server выполняется удаление всех установленных серверных и сервисных компонент SePlatform.Data Server. Предоставляется возможность выборочного удаления компонент посредством блокирования одной из частей в процессе удаления.

Чтобы удалить копии сервера (установленные нестандартным путем), выполните действия:

1. Разрегистрируйте копию сервера. В командной строке, находясь в папке `C:\Program Files\SePlatform\SePlatform.Server1\Server`, выполните команду:

```
SePlatform.ServerInstaller.exe /uninstall
```

2. Вручную удалите папку `C:\Program Files\SePlatform\SePlatform.Server1` со всеми файлами.

ОС Linux

Удаление SePlatform.Data Server выполняется штатным пакетным менеджером.

> ОБРАТИТЕ ВНИМАНИЕ
> Команда удаления выполняется только от суперпользователя **«root»**.

Удаление пакета `*.rpm` с помощью пакетного менеджера YUM:

```
yum remove seplatform.server
```

Удаление пакета `*.rpm` с помощью пакетного менеджера RPM:

```
rpm -e seplatform.server
```

Удаление пакета `*.deb` с помощью пакетного менеджера apt:

```
apt-get remove seplatform.server
```

Удаление пакета `*.deb` с помощью пакетного менеджера dpkg:

```
dpkg -r seplatform.server
```

4.6. Ручная настройка SePlatform.Data Server

Ручная настройка может потребоваться после установки SePlatform.Data Server для изменения стандартных параметров:

- расположение и имя бинарного файла конфигурации сервера;
- номер порта подключения к серверу;
- время выполнения автоматического резервного копирования;
- папка хранения резервных копий конфигурации сервера;
- папка хранения журналов работы модулей.

Параметры настройки сервера задаются в файле `SePlatform.Server.xml`, расположенном:

- в ОС Windows в папке `C:\Program Files\SePlatform\SePlatform.Server\Server`;
- в ОС Linux в директории `/opt/SePlatform/SePlatform.Server`.

Файл по умолчанию имеет следующий вид:

```
<?xml version="1.0" encoding="windows-1251"?>
<configuration>
  <install ServiceName="SePlatform.Server" ExeName="SePlatform.Server.exe">
    <ComServers>
      <OPCDA ProgID="SePlatform.OPCDAServer" CLSID="{28A2AD9C-C45E-4C6b-A0C3-
6E363F99CA72}" />
      <OPCAE ProgID="SePlatform.OPCAEServer" CLSID="{0CAEA48A-D7E6-44A4-85FD-
C27836727D07}" />
      <HDA ProgID="SePlatform.HDAServer" CLSID="{8002740A-886F-1488-2280-
42058D6D5CA8}" />
    </ComServers>
  </install>
  <Storage Filename="SePlatformServer.cfg" />
  <Connection Port="4572" />
  <Backup Path="..\Backups" Time="00:00" StorageDepth="14" />
  <Log Path="..\Logs" />
  <Dispatch Model="Default" />
  <Config ReadOnly="False" />
  <Instance ID="589745A2-5A75-4C4E-8540-B3795A0B2EF1" />
  <Culture LangID="ru-RU" />
  <MasterPassword
Cipher="icj+2PDqNPRFKuxi6q+XmyvU9QSEyS3iDtChbTKBIBWiuqo0Yslg7wOJz47D5UB02xeBurGx
CB0bPbZi8KSeX09CAZAIw8lyRYkPlCNhWA+NNq5X+ADZScDJmMEfDErpClJxCing7+2t3PiCjJWoxZAw
KpLWu1hmw9/QdikYQ9E" />
</configuration>
```

Допустимо изменение значений следующих атрибутов:

Родительский элемент	Атрибут	Описание
Storage	**Filename**	Полный путь или название файла конфигурации сервера
Connection	**Port**	Номер порта подключения к серверу
Backup	**Path**	Папка хранения автоматически создаваемых резервных копий конфигурации. Резервные копии по умолчанию сохраняются в папку `C:\Program Files\SePlatform\SePlatform.Server\Backups` с именами в формате: `[Год]_[Месяц]_[Дата] [номер копии] [имя пользователь].backup;`
Backup	**Time**	Время автоматического создания резервной копии текущей конфигурации

Родительский элемент	Атрибут	Описание
Backup	**StorageDepth**	Длительность хранения резервных копий конфигурации сервера, указывается в сутках. Диапазон допустимых значений от 0 до 90
Log	**Path**	Папка хранения журналов работы модулей сервера
MasterPassword	**Cipher**	Запись о пароле доступа к экземпляру сервера в зашифрованном виде, если пароль был задан в сервисном приложении Конфигуратор.

После внесения изменений в файл `SePlatform.Server.xml` сохраните и перезапустите службу SePlatform.Server в ОС Windows или сервис seplatform.server в Linux системах.

5. Конфигурирование SePlatform.Data Server

SePlatform.Data Server, как компонент Систэм Платформ, может быть сконфигурирован следующими способами:

- как отдельный компонент через сервисное приложение Конфигуратор;
- в составе проекта автоматизации с использованием SePlatform.Development Studio, при этом конфигурация SePlatform.Data Server компилируется из общей конфигурации проекта.

5.1. Принципы работы с конфигурацией

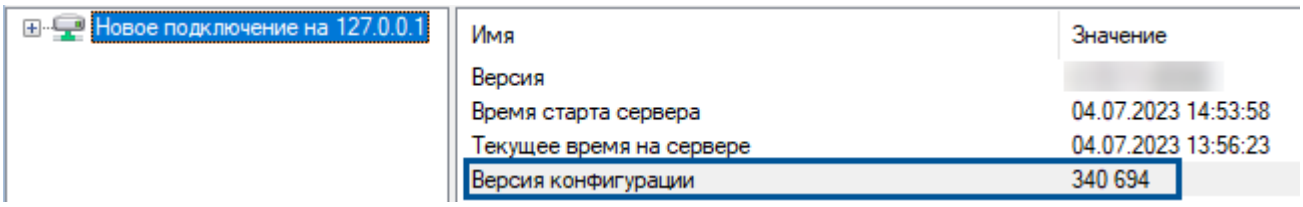
Конфигурация SePlatform.Data Server хранится в бинарном файле `SePlatformServer.cfg`. Файл расположен в папке/директории установки SePlatform.Data Server.

Конфигурация SePlatform.Data Server характеризуется:

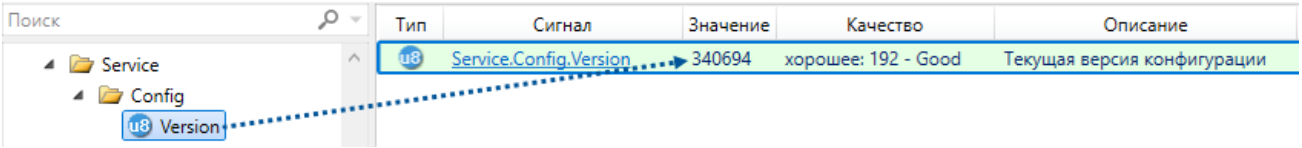
- номером версии конфигурации;
- идентификатором конфигурации.

Номер версии конфигурации увеличивается при любых изменениях конфигурации сервера. Текущий номер версии конфигурации можно узнать через:

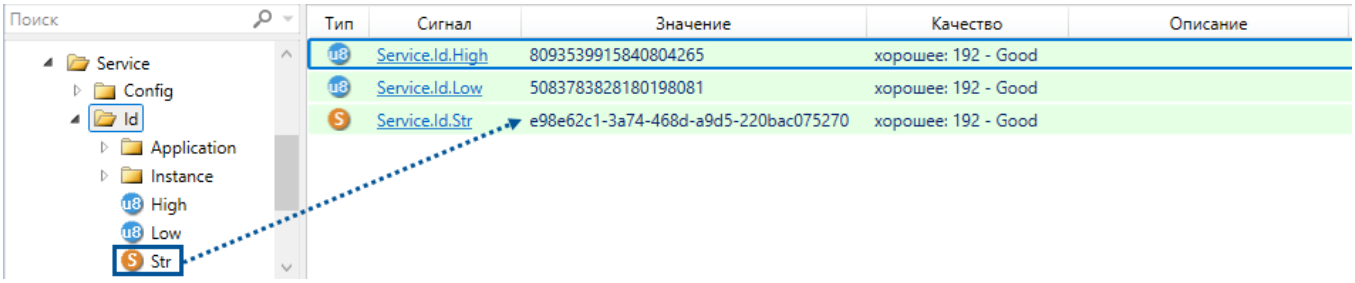
- сервисное приложение Статистика;



- сигнал с тегом **«Service.Config.Version»**, значение которого можно просмотреть любым OPC клиентом.



Конфигурация сервера имеет уникальный идентификатор, который содержится в сигнале с тегом **«Service.Id.Str»**. Идентификатор конфигурации также представлен в виде 16-байтового числа. Младшие 8 байт идентификатора содержатся в сигнале с тегом **«Service.Id.Low»**, старшие - в сигнале с тегом **«Service.Id.High»**. Данные сигналы можно смотреть любым OPC клиентом.



Также идентификатор можно узнать, открыв экспортированный файл конфигурации в текстовом редакторе.

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<!--файл конфигурации. Время создания: 9:00:27 05.07.2023, пользователь -
Administrator, компьютер - TESTER -->
<Configuration Id="e98e62c1-3a74-468d-a9d5-220bac075270" Format="1">
```

Конфигурация SePlatform.Data Server содержит:

- структура дерева сигналов - полный список сигналов, которые добавлены в конфигурацию;
- инициализирующие значения сигналов - значения, которые сигналы принимают при запуске сервера;
- модульный состав сервера - полный список модулей, которые добавлены в конфигурацию;
- конфигурационные параметры модулей и их значения;
- пароль доступа к серверу.

При старте сервера выполняется следующий порядок операций:

1. Конфигурация считывается из файла `SePlatformServer.cfg`.
2. Сигналам присваиваются инициализирующие значения.
3. Модули запускаются в нужной последовательности и согласно конфигурационным параметрам.
4. Модули достраивают дерево сигналов динамическими сигналами (стр. 35).

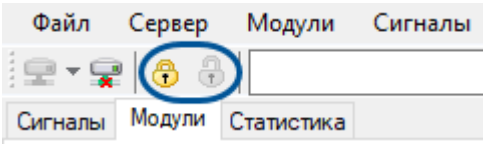
К адресному пространству сервера для работы с сигналами могут подключиться несколько OPC клиентов одновременно.

> ОБРАТИТЕ ВНИМАНИЕ
>
> Все изменения значений сигналов и свойств сигналов, которые произведены с помощью OPC клиента, не записываются в файл `SePlatformServer.cfg`. После перезапуска SePlatform.Data Server сигналы примут свои исходные инициализирующие значения.

Редактировать конфигурацию SePlatform.Data Server можно только с помощью сервисного приложения Конфигуратор. Изменения могут касаться состава дерева сигналов, списка модулей сервера, инициализирующих значений сигналов.

С конфигурацией можно работать в многопользовательском режиме. Для избежания конфликтов при совместной работе в Конфигураторе реализован механизм блокировки узлов и веток дерева сигналов или модулей. При редактировании любого узла конфигурации данный узел нужно блокировать для остальных пользователей. Также можно заблокировать целую ветку узлов. Далее работать с заблокированным узлом или веткой может только заблокировавший пользователь до снятия блокировки.

Блокировка веток и узлов конфигурации производится с помощью инструмента  на панели инструментов или командами главного меню **Блокировки** → **Заблокировать ветку/Заблокировать узел**. Снимается блокировка с помощью инструмента  на панели инструментов или командой главного меню **Блокировки** → **Снять блокировку**.



ПРИМЕЧАНИЕ

В OPC клиентах измененная конфигурация сервера будет доступна только после перезапуска SePlatform.Data Server.

Чтобы защитить конфигурацию сервера от потери данных, создавайте резервные копии текущей конфигурации (стр. 31). Сохраненные конфигурационные файлы могут быть обратно импортированы.

Экспортировать и импортировать конфигурацию сервера в формате `*.xmlcfg` и `*.csv` можно с помощью сервисного приложения Конфигуратор.

Чтобы экспортировать конфигурацию, в пункте меню **Файл** выберите Сохранить конфигурацию в файл....

Чтобы импортировать конфигурацию из файла, в пункте меню **Файл** выберите **Загрузить конфигурацию из файла...**.

Файлы конфигурации формата `*.xmlcfg` доступны для просмотра с помощью текстового редактора. Файлы формата `*.csv` открываются с помощью MS Excel и имеют вид структурированной таблицы.

ОБРАТИТЕ ВНИМАНИЕ

После всех изменений конфигурации или после импорта конфигурации необходимо перезапустить SePlatform.Data Server.

5.2. Принципы модульного строения

Многие возможности SePlatform.Data Server реализуются модулями - специальными библиотеками. Так, весь обмен данными по стандартным протоколам - МЭК 870-5, MODBUS, SNMP и другим - реализуется как раз с использованием модулей.

Одни модули могут получать данные и сохранять их как значения сигналов, другие модули могут предоставлять потребителям хранящиеся в сигналах данные. Например, модуль Modbus TCP Master запрашивает у станции Modbus значение какого-нибудь технологического параметра и сохраняет его в виде значения некоторого сигнала.

Один и тот же сигнал может одновременно использоваться разными модулями. Например, модуль Modbus TCP Master может записать значение параметра в сигнал, а затем модуль OPC UA передаст значение этого сигнала в SePlatform.HMI.

5.3. Настройка общих параметров модулей

После формирования списка модулей настройте параметры их работы. Все модули обладают общим и уникальным наборами параметров. Уникальные параметры узла конфигурации модуля содержатся в группе **Дополнительные** и настраиваются для каждого модуля индивидуально.

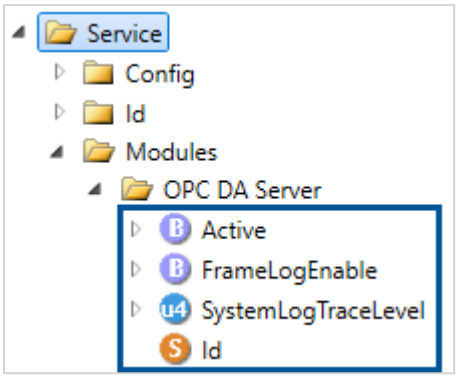
1. Общие	
Имя модуля	OPC DA Server
Идентификатор модуля	OPC DA Server
Активность	Да
Уровень трассировки в журнал приложений	Информационные сообщения
Вести журнал работы модуля	Нет
Размер журнала работы модуля, МБ	100
Количество дополнительных журналов работы	10
2. Дополнительные	
Максимальный размер очереди уведомлений	500000

Общие параметры узла конфигурации модуля содержатся в группе **Общие**:

Параметр	Описание
Имя модуля	Название модуля
Идентификатор модуля	Идентификатор модуля в SePlatform.Data Server
Активность	Активность модуля: › **«Да»** - модуль запущен › **«Нет»** - модуль остановлен
Уровень трассировки в журнал приложений	Типы сообщений, которые фиксируются в журнал приложений: › **«Предупреждения и аварийные сообщения»** - логические ошибки, ошибки работы модуля SePlatform.Data Server. Предупреждения содержат некритичные ошибки. Аварийные сообщения информируют об ошибках, которые влияют на работоспособность SePlatform.Data Server; › **«Информационные сообщения»** - сообщения, которые показывают основную информацию о работе модуля; › **«Отладочные сообщения»** - сообщения, которые наиболее детально отражают информацию о работе модуля. Вышестоящий уровень входит в состав нижестоящего. Если установлен уровень **«Информационные сообщения»**, то в журнал фиксируются **«Предупреждения и аварийные сообщения»** и **«Информационные сообщения»**
Вести журнал работы модуля	Параметр, показывающий ведется ли запись сообщений о работе модуля в журнал работы модуля: › **«Да»** - сведения о работе модуля сохраняются в журнал › **«Нет»** - журнал работы модуля не ведётся
Размер журнала работы модуля, МБ	Размер файла журнала работы модуля в мегабайтах. При достижении максимального размера создается новый файл, копия старого файла хранится на рабочем диске
Количество дополнительных журналов работы	Количество файлов заполненных журналов работы модуля. Минимальное значение параметра равно **«1»**. Максимальное количество файлов журнала равно **«255»**.

Значения общих параметров таких как **Активность** (**«Active»**), **Вести журнал работы модуля** (**«FrameLogEnable»**), **Идентификатор модуля** (**«Id»**) и **Уровень трассировки в журнал приложений** (**«SystemLogTraceLevel»**), вы можете посмотреть в любом OPC DA клиенте по тегу:

```
Service.Modules.Имя модуля.Имя параметра
```



Ведение журнала работы для каждого модуля настраивается при конфигурировании модуля. Для каждого модуля создается отдельный файл:

- в ОС Windows в папке `C:\Program Files\SePlatform\SePlatform.Server\Logs`;
- в ОС Linux в директории `/opt/SePlatform/Logs`.

Ведение журнала работы для каждого модуля настраивается при конфигурировании модуля. Для каждого модуля создается отдельный файл. Расположение файлов журналов работы указывается в файле `SePlatform.Server.xml` в атрибуте **Path** элемента **Log** (по умолчанию `Logs` в папке/директории установки SePlatform.Data Server).

Имя файла совпадает с названием модуля. Расширение файлов журнала `*.aplog`. Размер файла журнала работы модуля устанавливается вручную в конфигурации каждого модуля и по умолчанию равен **«10»** Мбайт. При заполнении заданного размера файла журнала создается другой файл журнала (количество дополнительных журналов зависит от количества журналов, настроенных при конфигурации модуля), в который сохранятся все записи текущего журнала. Текущий журнал, в свою очередь, очистится, и запись работы модуля возобновится в нем. При последующем заполнении текущего файла журнала создается еще один файл журнала, если в конфигурации модуля он был предусмотрен, либо уже созданный файл журнала будет перезаписан. После перезагрузки SePlatform.Data Server записи в журнале работы модуля не удаляются, а добавляются к уже имеющимся в журнале.

ОС Windows

Для просмотра журналов работы модулей SePlatform.Data Server используется сервисное приложение Просмотрщик лога кадров.

Чтобы просмотреть журнал работы приложений с заданным уровнем трассировки сообщений, воспользуйтесь приложением Service - LogViewer.

ОС Linux

Команда просмотра журнала работы:

```
journalctl -u seplatform.server
```

5.4. Пример ручного создания конфигурации

В зависимости от задач, которые предстоит выполнять серверу, определяется модульный состав SePlatform.Data Server. Ниже приведены несколько примеров:

- если необходимо предоставлять данные клиентам по спецификации OPC DA, то в конфигурацию необходимо добавить модуль OPC DA Server (только в ОС Windows);
- если необходимо опрашивать некое устройство по спецификации Modbus TCP, то в конфигурацию необходимо добавить модуль Modbus TCP Master;
- если необходимо записывать оперативные данные в сервер истории, то в конфигурацию необходимо добавить модуль History Module.

Шаги по созданию минимальной конфигурации:

1. Добавьте нужные модули в конфигурацию сервера в зависимости от ваших задач.
2. Чтобы модули начали функционировать, переведите общий параметр модулей **Активность** в значение **«Да»**.
3. Настройте общие и дополнительные параметры модулей (стр. 24).
4. Чтобы сервер оперировал данными потребуются сигналы, которые являются контейнерами для данных. Создайте нужное количество сигналов определенного типа (стр. 34).
5. Чтобы более подробно описать характеристики сигналов или поставить сигналы на обслуживание какому-либо модулю, добавьте для сигналов свойства (стр. 35). Примеры свойств сигналов:
   - чтобы детально описать сигнал, определите для него свойство **`101 (Description)`**;
   - если сигнал отражает значение физического параметра, то для него можно определить свойство **`100 (Eunit)`**, которое будет отражать единицы измерения;
   - чтобы поставить сигнал на обслуживание некоторому модулю, настройте привязку через строковое свойство **`5000 (Address)`**;
   - чтобы настроить сигналу сохранение значений в историю, добавьте свойства **`9001`** и **`9002`**.

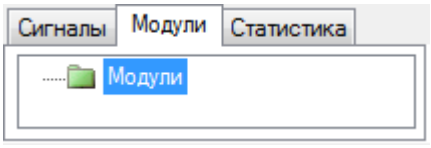
ОБРАТИТЕ ВНИМАНИЕ

Чтобы изменения вступили в силу, после всех изменений конфигурации необходимо перезапустить SePlatform.Data Server.

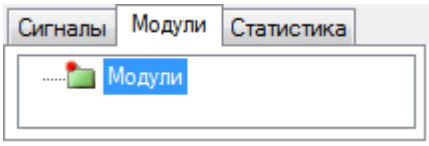
Пример добавления и активация модуля

Чтобы OPC DA клиенты получили возможность подключаться к SePlatform.Data Server и запрашивать от него данные по спецификации OPC DA, необходимо добавить и настроить модуль OPC DA Server:

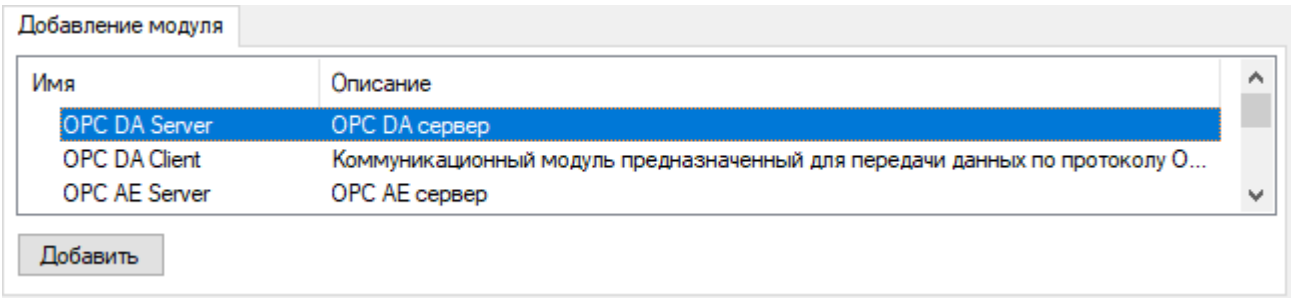
1. Перейдите на закладку **Модули** в левой области Конфигуратора.



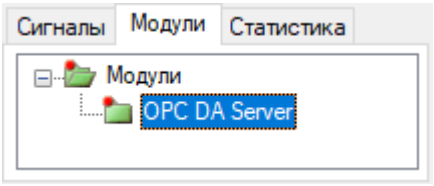
2. Заблокируйте корень дерева модулей с помощью инструмента или команды **Заблокировать ветку** меню **Блокировки**. Вид заблокированного дерева модулей представлен на рисунке ниже.



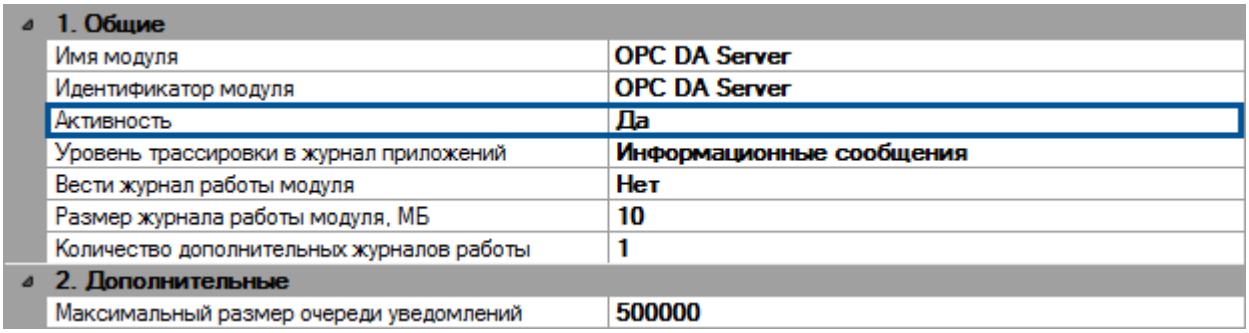
3. Из списка модулей добавьте модуль OPC DA Server.



Добавленный модуль появится в дереве модулей.



4. Активируйте модуль OPC DA Server. На закладке **Параметры узла конфигурации модуля** в группе **Общие** для параметра **Активность** установите значение **«Да»**.



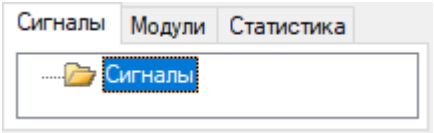
5. Разблокируйте корень дерева модулей с помощью инструмента  или команды **Снять блокировку** меню **Блокировки**. В появившемся окне сохраните внесенные изменения.

Пример добавления сигнала и его свойств

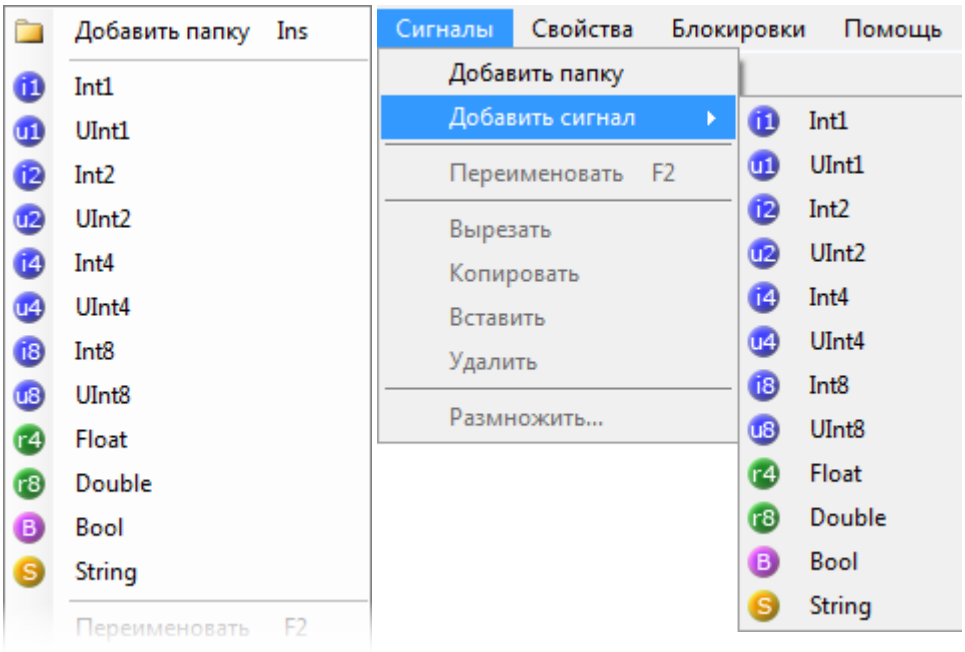
В рамках примера будет добавлен сигнал типа UInt2 со свойствами **`2 (Value)`**, **`3 (Quality)`**, **`101 (Description)`**.

Чтобы добавить сигнал в конфигурацию сервера:

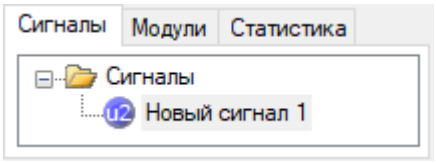
1. Перейдите на закладку **Сигналы** в левой области Конфигуратора.



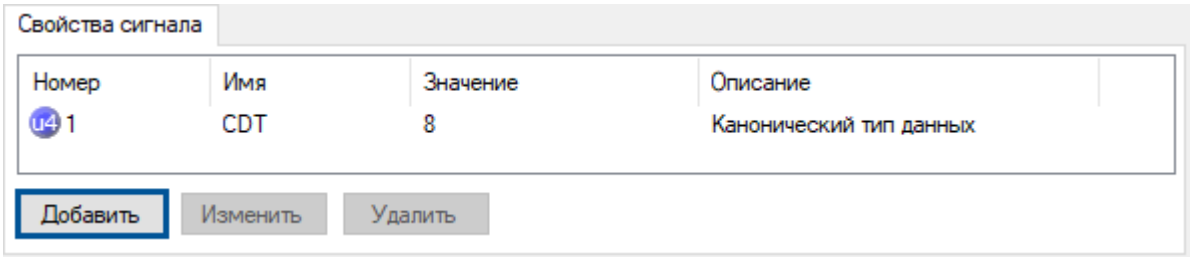
2. Из контекстного меню или меню **Сигналы** выберите необходимый тип сигнала.



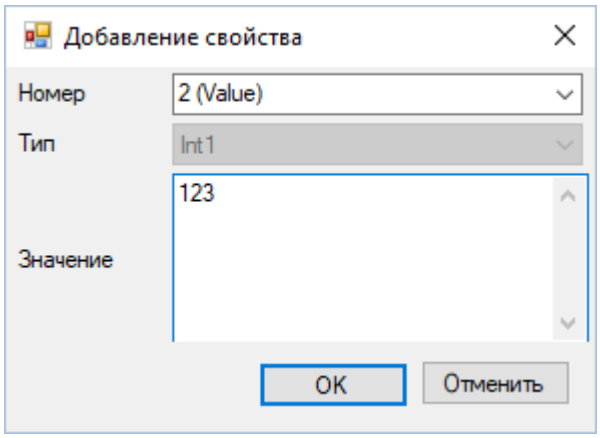
Добавленный сигнал появится в дереве сигналов.



3. Чтобы добавить свойства сигналу, нажмите **Добавить** в области **Свойства сигнала**.



4. В окне **Добавление свойств** укажите **Номер** и **Значение** для нового свойства. Поле **Тип** заполнится автоматически. Настройка свойства **2 (Value)** показана на рисунке ниже.



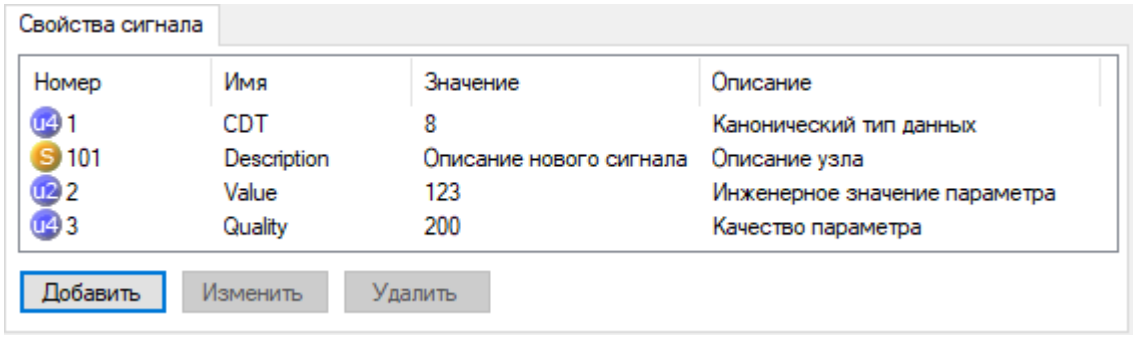
Настройка для свойства **3 (Quality)**:

- Номер - **«3 (Quality)»**;
- Значение - **«200»**.

Настройка для свойства **101 (Description)**:

- Номер - **«101 (Description)»**;
- Значение - **«Описание Новый сигнал 1»**.

Область **Свойства сигнала** после добавление свойств сигналу показана на рисунке ниже.

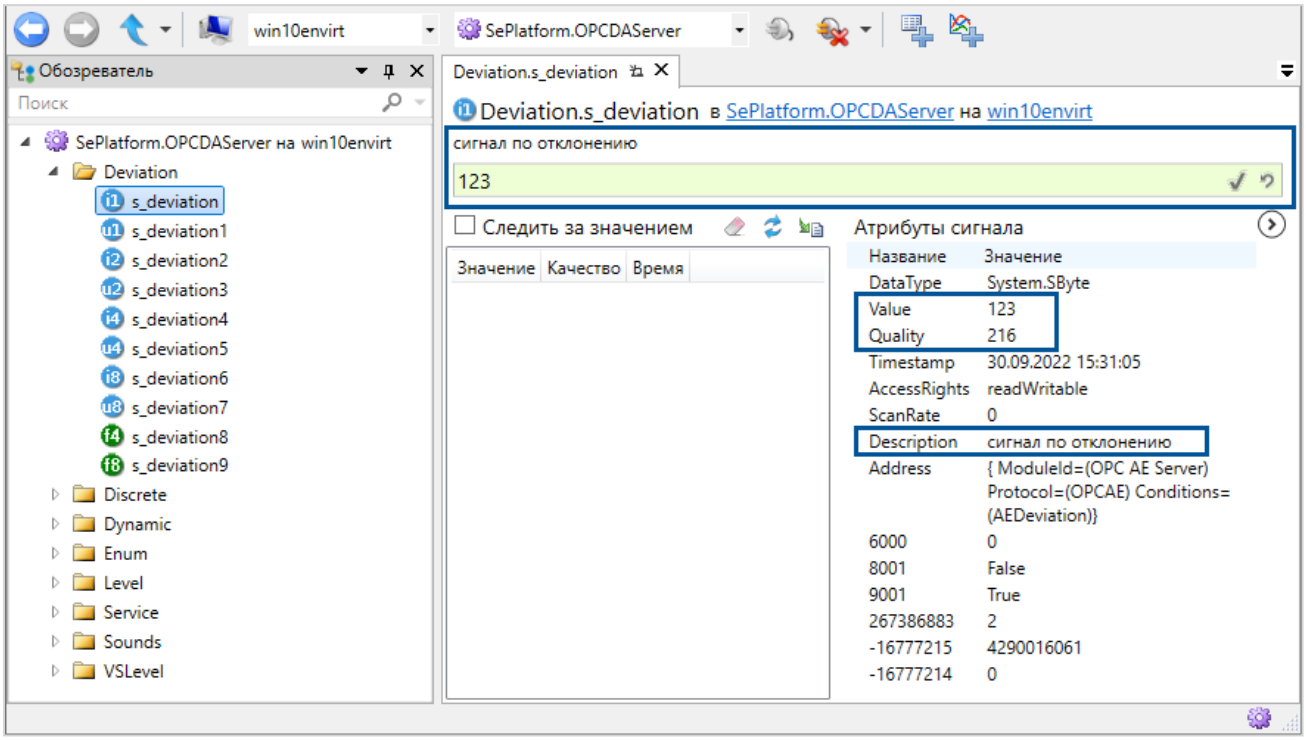


Пример просмотра значений сигналов OPC DA клиентом

В рамках примера будут просмотрены значения добавленных сигналов OPC клиентом Service - OPCExplorer.

Чтобы просмотреть значения сигналов:

1. Подключитесь OPC клиентом к OPC DA серверу SePlatform.Data Server.
2. Перейдите в свойства сигнала **«Новый сигнал 1»**.

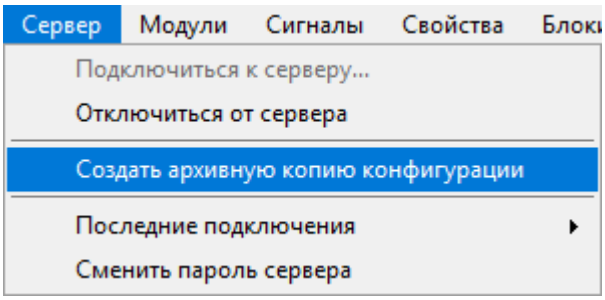


5.5. Резервное копирование конфигурации сервера

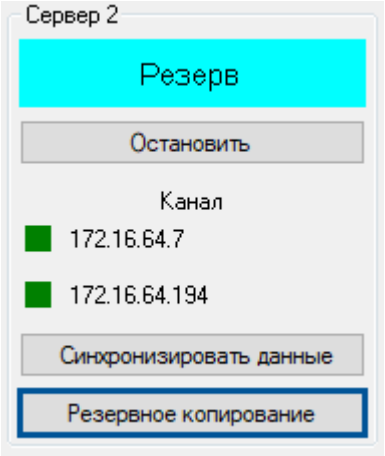
Создание резервной копии конфигурации SePlatform.Data Server возможно выполнить следующими способами:

- автоматически. В автоматическом режиме резервное копирование выполняется в соответствии с настройками расписания, указанными в файле настроек сервера `SePlatform.Server.xml` (стр. 19).
- вручную через приложение Конфигуратор и приложение Управляющий.

Чтобы создать резервную копию конфигурации через сервисное приложение Конфигуратор, в окне приложения выберите команду **Сервер** → **Создать архивную копию конфигурации**.



Чтобы создать резервную копию конфигурации сервера с помощью сервисного приложения Управляющий, в окне приложения нажмите кнопку **Резервное копирование**.



Настройки сохранения резервных копий в ручном и в автоматическом режиме задаются в файле настроек сервера `SePlatform.Server.xml` теге **<Backup>**:

- **Path** - название каталога, в который сохраняются резервные копии;
- **Time** - время автоматического выполнения резервного копирования;
- **StorageDepth** - длительность хранения резервных копий на диске. Указывается в сутках, значение по умолчанию **«14»** суток.

Чтобы восстановить конфигурацию SePlatform.Data Server из резервной копии:

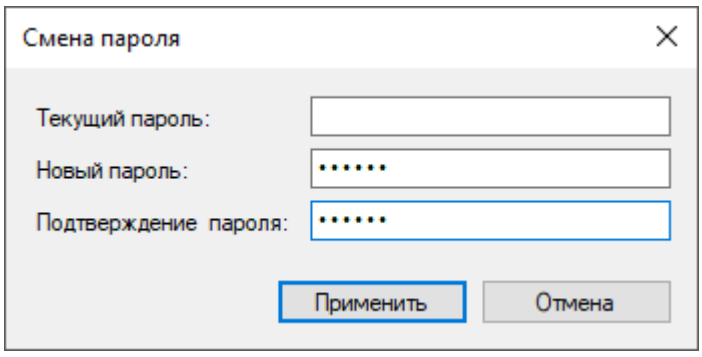
- файлу резервной копии задайте имя, указанное в файле `SePlatform.Server.xml` (в папке/директории установки SePlatform.Data Server) в атрибуте **Filename** элемента **Storage** (по умолчанию `SePlatformServer.cfg`);
- замените текущий файл конфигурации SePlatform.Data Server.

5.6. Защита от несанкционированного доступа

Установка пароля доступа к SePlatform.Data Server предотвращает следующие несанкционированные действия:

- управление сервером или резервной парой серверов через сервисное приложение Управляющий;
- модификацию конфигурации через сервисное приложение Конфигуратор;
- просмотр статистической информации через сервисное приложение Статистика;
- обмен данными с SePlatform.AccessPoint.

Операции с паролем доступа (создание/изменение) выполняются в окне **Смена пароля**, которое запускается командой меню **Сервер** → **Сменить пароль сервера** из сервисного приложении Конфигуратор.



После первого подключения к серверу (не требует ввода пароля) следует назначить пароль через окно **Смена пароля**.

ОБРАТИТЕ ВНИМАНИЕ

Пароль хранится в файле `SePlatform.Server.xml` в зашифрованном виде.

5.7. Шифрование паролей

Пароль доступа к серверу, а также пароли, указываемые в параметрах некоторых модулей, хранятся в зашифрованном виде. Для шифрования паролей используется ассиметричный алгоритм шифрования RSA с ключом размером 1024 бита:

- введённый пароль шифруется с помощью открытого ключа и сохраняется в зашифрованном виде;
- сохранённый пароль расшифровывается только когда он нужен в открытом виде (например, при сравнении паролей). Расшифровка выполняется с помощью закрытого ключа. Закрытый ключ зашифрован алгоритмом blowfish и в открытом виде нигде не хранится.

5.8. Контроль целостности конфигурации и БД

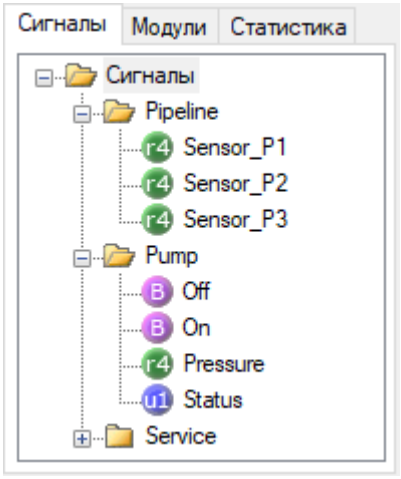
При запуске SePlatform.Data Server выполняет проверку целостности конфигурации, а в процессе работы - проверку целостности БД реального времени в части доступности данных. В случае выявления ошибок информация о них выводится в системный журнал или журналы работы модулей SePlatform.Data Server. Для получения подробной информации о сообщениях проверки целостности конфигурации и БД обратитесь в техническую поддержку ООО «СИСТЭМ СОФТ» по электронной почте `support@systemesoft.ru` или на сайте https://www.systemesoft.ru/.

6. Сигналы SePlatform.Data Server

Для сбора данных о состоянии технологического процесса и для доставки команд пользователя оборудованию SePlatform.Data Server использует сигналы. Сигнал - это переменная определённого типа: целое или вещественное число, логическая переменная, строка.

Сигнал может хранить значение некоторого технологического параметра. Например, давление в трубопроводе, состояние компрессора или сообщение системы.

SePlatform.Data Server хранит сигналы в виде дерева. Для удобства его структуру можно менять.



6.1. Типы данных

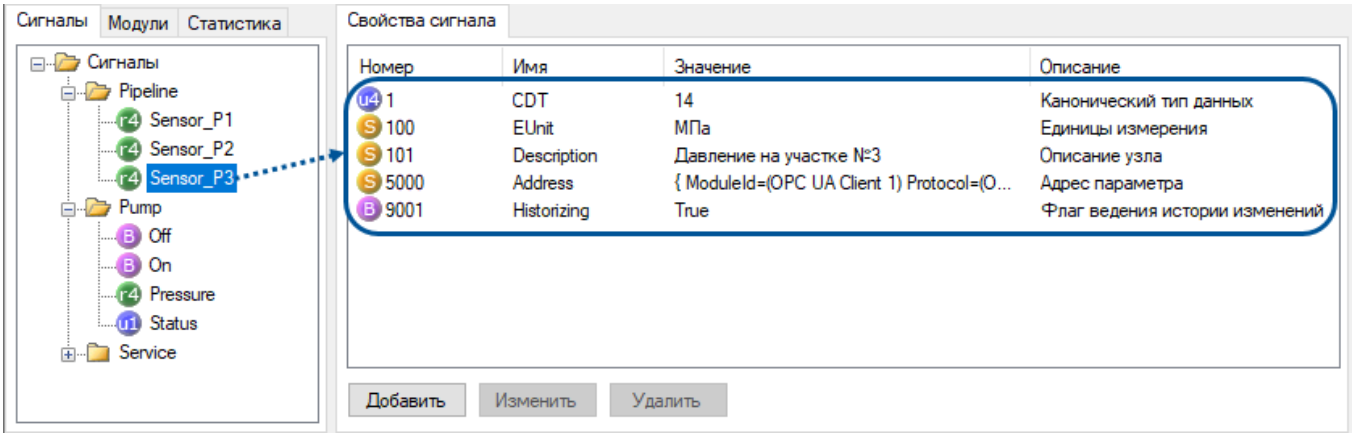
Тип данных сигнала определяет, какое множество значений может принимать технологический параметр, описываемый данным сигналом. Возможные типы данных SePlatform.Data Server показаны ниже.

Тип	Описание	Допустимые значения
Int1	Знаковое целое 1 байт	[-128; 127]
UInt1	Беззнаковое целое 1 байт	[0; 255]
Int2	Знаковое целое 2 байта	[-32 768; 32 767]
UInt2	Беззнаковое целое 2 байта	[0; 65 535]
Int4	Знаковое целое 4 байта	[-2 147 483 648; 2 147 483 647]
UInt4	Беззнаковое целое 4 байта	[0; 4 294 967 295]
Int8	Знаковое целое 8 байт	[-9 223 372 036 854 775 808; 9 223 372 036 854 775 807]
UInt8	Беззнаковое целое 8 байт	[0; 18 446 744 073 709 551 615]

Тип	Описание	Допустимые значения
Float	Значение с плавающей запятой 4 байта	$[\pm 1.5\times10^{-45}; \pm 3.4\times10^{38}]$. Точность 6-9 цифр
Double	Значение с плавающей запятой 8 байт	$[\pm 5.0\times10^{-324}; \pm 1.7\times10^{308}]$. Точность 15-17 цифр
Bool	Логическое значение	true, false
String	Текстовая строка в кодировке UTF16	до 2 миллиардов знаков, каждый знак занимает 16 бит (2 байта)

6.2. Свойства сигналов

Кроме значения технологического параметра, сигнал хранит дополнительную информацию о параметре. Например, единицы измерения значения (МПа, мм), описание (**«Давление в трубопроводе»**, **«Уровень в резервуаре»**). Эта информация хранится в свойствах сигнала.



Полный список свойств сигналов SePlatform.Data Server приведен в приложении (стр. 60).

6.3. Статические и динамические сигналы

Все сигналы сервера делятся на 2 типа:

- Статические - сохраняются в файл конфигурации `SePlatformServer.cfg` (стр. 22). Инициализирующие значения статистических сигналов задаются и корректируются с помощью приложения Конфигуратор и SePlatform.Development Studio. Заданные значения устанавливаются сигналам и свойствам при старте сервера. В случае изменения значений сигналов и свойств с помощью OPC DA клиентов, внесенные изменения действуют только до перезапуска сервера. OPC DA клиенты могут подписываться только на именованные свойства сервера. Сервер позволяет подписаться и изменять свойства **2**, **3**, **4** только с использованием модуля Write VQT Module.

› Динамические - создаются ядром сервера или модулями сервера. Динамические сигналы не сохраняются в конфигурации сервера. Служебные сигналы, созданные модулями и ядром сервера, позволяют наблюдать за работой сервера или конкретного модуля. Просмотр и изменение сигналов и свойств возможен только с помощью OPC DA клиентов. Изменения значений динамических сигналов и свойств действуют только до перезагрузки сервера. После перезапуска сервера сигнал или свойство принимает исходное значение. Например, динамическим свойством является свойство **`5002 (RawValue)`**, которое создается ядром при взятии сигнала на обслуживание.

6.4. Возможные значения качества сигналов

Достоверность значения технологического параметра определяется исходя из значения свойства **`3 (Quality)`**. Качество сигнала является индикатором возможности использования значения сигнала при оценке оперативной ситуации.

Инициализирующие значения качества сигналов настраиваются в конфигурации SePlatform.Data Server. В случае если качество не было задано на этапе конфигурирования сигналов, при старте сервера ядро устанавливает всем сигналам качество OUT_OF_SERVICE. Указанное значение качества говорит о том, что сигнал не поставлен на обслуживание ни одним модулем. Далее сигналы передаются на обслуживание модулям и качество выставляется обслуживающими модулями. Качество сигнала может изменяться при пересчете значения сигнала.

Качество сигналов можно просмотреть, подключившись к SePlatform.Data Server с помощью OPC DA клиента. Значение качества сигнала может быть установлено модулями сервера на основании данных и алгоритмов их работы в соответствии со спецификацией OPC DA 2.05a.

Значение сигнала достоверно, если качество имеет одно из следующих значений:

Значение	**Описание**	**Возможные причины**
GOOD (192)	Значение получено от устройства (источника данных) с хорошим качеством	Значение установлено одним из модулей или задано в конфигурации сервера
CALCULATED (200)	Значение получено в результате вычислений	Качество всех аргументов достоверное (>=192) и не возникло переполнений
LOCAL_ OVERRIDE (216)	Значение изменено пользователем вручную	Значение изменено пользователем по OPC

Значение сигнала недостоверно и должно игнорироваться, если качество имеет одно из следующих значений:

Значение	**Описание**	**Возможные причины**
BAD (0)	Данные получены с плохим качеством	В регистре статуса сигнала старший бит не равен нулю
CONFIG_ERROR (4)	Неверно заданы параметры сигнала	Неверный адрес сигнала. Например: одно из полей структуры адреса задано неверно. Расширенную информацию можно получить из журнала работы модуля

Значение	Описание	Возможные причины
NOT_ CONNECTED (8)	Модуль, обслуживающий сигнал, не подключен к источнику данных	Соединение с подчиненной станцией не установлено. Например, последовательный порт, через который поступает значение сигнала, не открыт
DEVICE_ FAILURE (12)	Модулю, обслуживающему сигнал, не удалось подключиться к источнику данных	Не удалось открыть устройство, осуществляющее прием/передачу данных. Например: не удалось открыть последовательный порт, через который получается значение сигнала
SENSOR_ FAILURE (16)	Устройство (источник данных), с которого должно быть получено значение сигнала, неисправно. От устройства получено значение сигнала, выходящее за допустимые пределы	Датчик неисправен
LAST_KNOWN (20)	Связь с устройством (источником данных), поставляющим значение сигнала, потеряна. Отображается последнее известное значение параметра	Отсутствует связь с устройством
COMM_FAILURE (24)	Не удалось установить связь с устройством (источником данных), от которого должно быть получено значение сигнала	Отсутствует связь с устройством, который должен поставлять сигнал; Например, КП номер 3, связанный с сигналом, отсутствует на связи
OUT_OF_ SERVICE (28)	Сигнал не обслуживается	Сигнал не поставлен на обслуживание ни одним активным (запущенным) модулем. Например: адрес сигнала содержит неверный тип протокола (ни один из запущенных модулей не поддерживает указанный протокол)
WAITING_FOR_ INITIAL_DATA (32)	С момента инициализации SePlatform.Data Server значение, качество и метка времени сигнала не изменялись. При этом сигналу установлено значение **«0»**	Данные от контроллера не приходят
UNCERTAIN (64)	Данные не получены	Связь с устройством (источником данных), поставляющим значение сигнала, установлена, но данные еще не получены
EGU_EXCEEDED (84)	(Engineering Units Exceeded). Возвращаемое значение выходит за границы, определенные для этого параметра	Значение сигнала вышло за рамки при пересчете

Значение	Описание	Возможные причины
SUB_NORMAL (88)	Значение вычислено из множества значений и не все из них имеют качество GOOD	Значение сигнала вычислено на основе нескольких сигналов и не все сигналы имеют достоверное качество

6.5. Адресация сигналов

Основным назначением SePlatform.Data Server является обмен технологическими данными со сторонними OPC серверами по спецификации OPC DA v.2.05a, а также с устройствами телемеханики и станционной автоматики по следующим коммуникационным протоколам и технологиям:

- ГОСТ Р МЭК 60870-5-104;
- Modbus TCP;
- Modbus RTU;
- SNMP;
- SQL-запросы;
- ICMP.

Сбор, передача данных и выдача управляющих воздействий осуществляется с помощью технологических сигналов SePlatform.Data Server. Для привязки сигнала к коммуникационным модулям SePlatform.Data Server используется свойство **`5000 (Address)`**. Формат адреса сигнала определяется используемым протоколом передачи данных. Каждый сигнал может быть привязан к одному или нескольким коммуникационным модулям SePlatform.Data Server. В случае связи с несколькими модулями количество адресов сигнала совпадает с количеством модулей.

Адрес сигнала содержится в свойстве **`5000 (Address)`**. Указанное свойство может содержать данные только строкового типа. Адрес сигнала является статическим свойством и создается пользователем вручную. Синтаксис установки свойства **`5000 (Address)`** приведен в документах на конкретные модули.

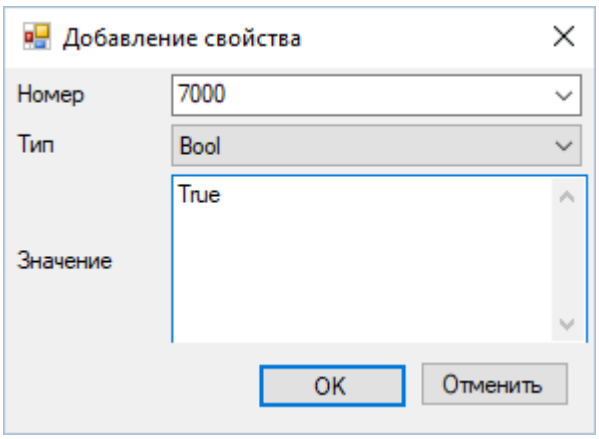
Создание и настройка адресов технологических сигналов SePlatform.Data Server выполняется с помощью сервисного приложения Конфигуратор.

6.6. Настройка ведения детального журнала по выбранному сигналу

Данная функция позволяет вести детальный журнал изменений сигналов. По умолчанию, ведение детального журнала по изменениям сигналов отключено. Ведение детального журнала настраивается в сервисном приложении Конфигуратор. Ведение детального журнала изменений сигналов реализовано в следующих модулях:

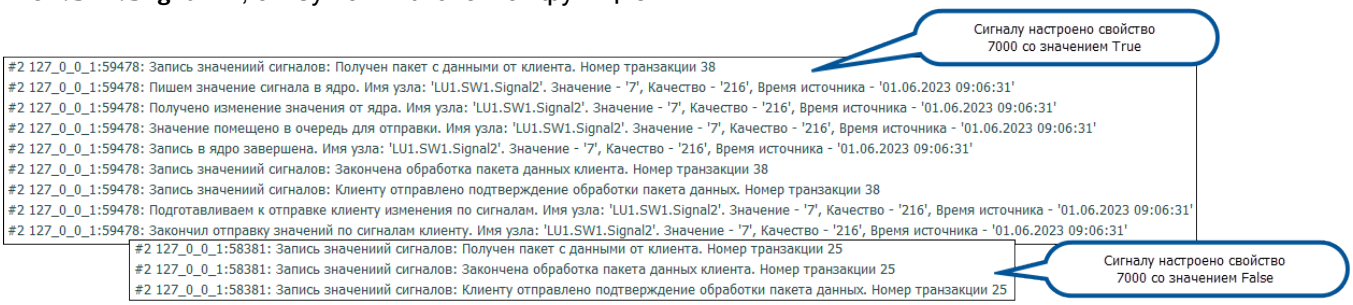
- IEC-104 Master;
- TCP Server Module;
- HUB Module;
- OPC DA Server;
- OPC DA Client.

Чтобы настроить сигналу ведение детального журнала, в окне **Добавление свойства** введите номер свойства - **`7000`**, тип данных - bool и значение - **«`True`»**. Чтобы отключить функцию ведения детального журнала, в окне **Добавление свойства** укажите значение **«`False`»** или удалите свойство **`7000`**.



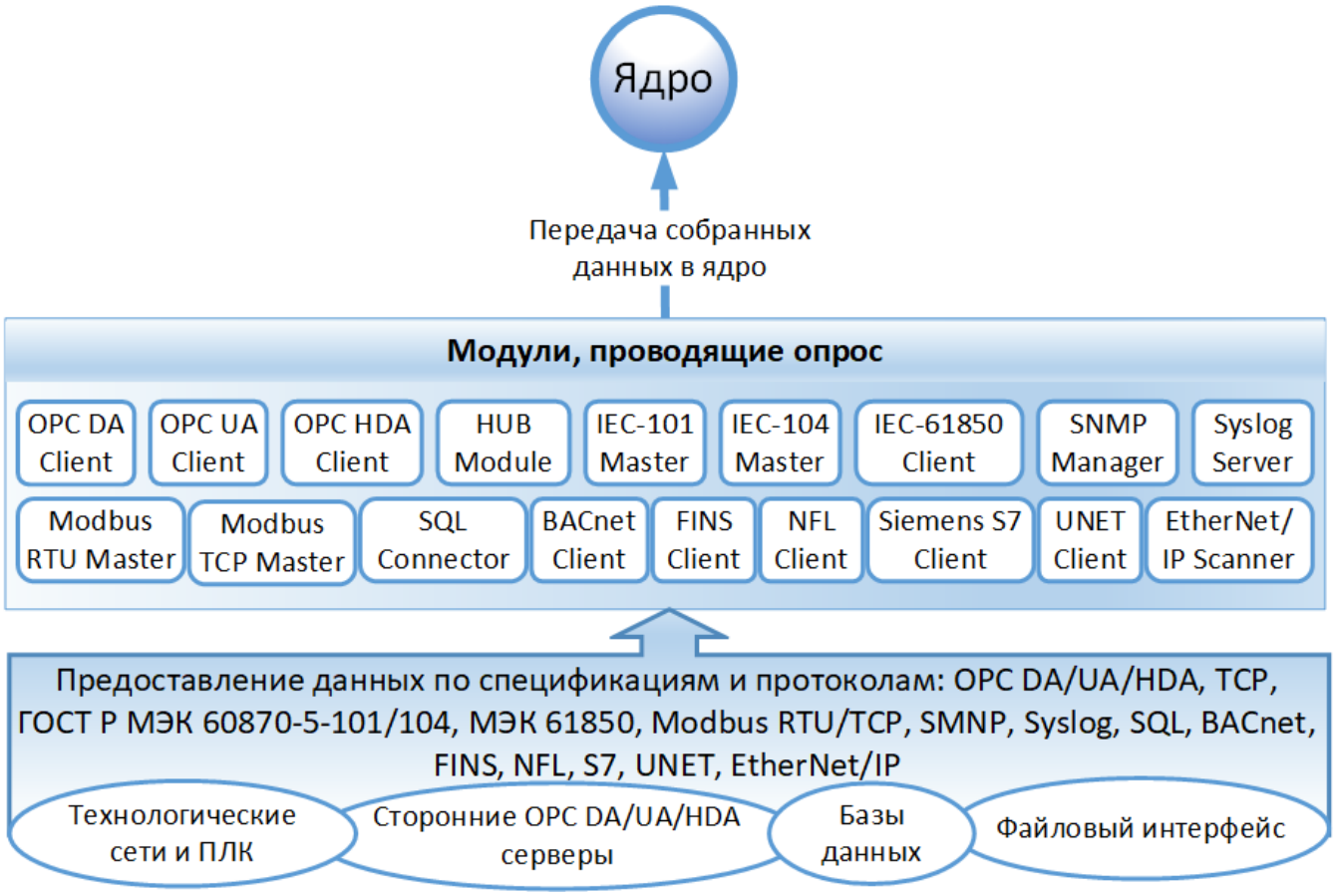
Чтобы просмотреть детальный журнал изменений по выбранному сигналу, используйте сервисное приложение Просмотрщик лога кадров.

На рисунке ниже показаны фрагменты журнала работы модуля TCP Server Module. На рисунке сверху показан лог с включенной функцией ведения детального журнала изменений для сигнала «**LU1.SW1.Signal2**», снизу - с выключенной функцией.



7. Сбор данных

Сбор технологических данных происходит при опросе коммуникационных устройств, ПЛК и сторонних серверов. Схема сбора данных и передачи их в ядро SePlatform.Data Server показана на рисунке ниже.



SePlatform.Data Server способен собирать данные по различным коммуникационным протоколам и спецификациям.

Чтобы настроить SePlatform.Data Server на сбор данных:

1. Добавьте в состав конфигурации модули, которые соответствуют выбранным протоколам/спецификациям, и активируйте их.
2. Настройте модули на сбор данных. Как настроить модули см. в документе на соответствующий модуль.

8. Логическая обработка данных

Логическая обработка данных, собранных с коммуникационных устройств, ПЛК и сторонних серверов - это одна из основных задач SePlatform.Data Server.

Логические вычисления, такие как пересчет значений сигналов (стр. 41) и перекладка данных между сигналами (стр. 48), выполняются ядром SePlatform.Data Server и доступны без использования модуля логики. Для прочих вычислений в составе конфигурации SePlatform.Data Server необходим модуль Logics Module.

Основные отличия логической обработки данных в ядре и модулем логики:

1. Пересчет значений сигналов и перекладка данных между сигналами не реализуется с помощью модуля вычислений.
2. Простые логические вычисления, пересчет значений сигналов и перекладка данных между сигналами, происходят синхронно, а логические вычисления с помощью модуля логики - асинхронно.

Синхронный подход обработки данных выражается в том, что пока ядро сервера производит логическую обработку данных, оно не сможет выполнять больше никаких других функций. При асинхронном подходе обработки данных модуль логики выполняет вычисления всех значений сигналов параллельно без участия ядра сервера, в ядро записывается уже обработанное модулем значение сигнала.

Чтобы настроить логическую обработку данных в ядре SePlatform.Data Server, добавьте в сигнал свойства, отвечающие за пересчет значений или перекладку данных между сигналами.

8.1. Пересчет значений сигналов

Данные, поступающие с подчиненных станций в SePlatform.Data Server, представляют собой физические величины. Для корректной обработки входящих значений параметров физические величины, полученные с физических устройств измерения, переводятся в инженерные. Для подачи управляющих воздействий с пункта управления инженерные величины переводятся в физические. Пересчет выполняется с помощью встроенной в ядро SePlatform.Data Server функции пересчета.

Физическое значение параметра записывается в свойство сигнала **`5002 (Raw Value)`**. Инженерное значение параметра содержится в свойстве сигнала **`2 (Value)`**. Свойство **`5002 (Raw Value)`** создается ядром SePlatform.Data Server при постановке сигнала на обслуживание коммуникационным модулем.

Функция пересчета физического значения в инженерное активируется, если изменилось значение свойства **`5002 (Raw Value)`** или значение любого из свойств пересчёта (стр. 42) (например, с помощью OPC клиента). Полученное в процессе пересчета результирующее значение будет записано в свойство **`2 (Value)`**. Пересчет инженерного значения в физическое производится при изменении свойства **`2 (Value)`**. Результат пересчета будет записан в свойство **`5002 (Raw Value)`**.

В SePlatform.Data Server возможен пересчет следующими методами:

- Линейный пересчет (стр. 42);
- Линейный пересчет с изломом (стр. 46);
- Инверсия битовых сигналов (стр. 48).

8.1.1. Настройка пересчета значений сигналов

Для настройки пересчета необходимо создать набор свойств, в которых задаются параметры пересчета. Для каждого вида пересчета предназначен определенный набор свойств. Список свойств пересчета представлен в таблице ниже.

ID	Тип	Короткое имя	Описание
5100	double	RecalcRawLow	Нижняя граница физического значения
5101	double	RecalcRawMiddle	Граница излома физического значения
5102	double	RecalcRawHigh	Верхняя граница физического значения
5103	double	RecalcValLow	Нижняя граница инженерного значения
5104	double	RecalcValMiddle	Граница излома инженерного значения
5105	double	RecalcValHigh	Верхняя граница инженерного значения
5106	bool	RecalcTruncate	Усекать значение по границе пересчета и добавлять в качество флаги усечения (LIMIT_LOW или LIMIT_HIGH)
5107	bool	RecalcSetFailureQuality	При усечении по границе пересчета выставлять SENSOR_FAILURE
5108	bool	RecalcInvert	Инвертировать логическое значение. Действует только для сигналов с типом bool.

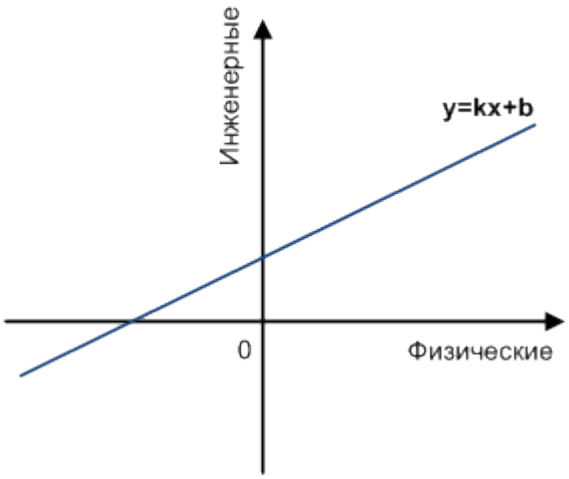
Если сигнал имеет свойство **`5002 (Raw Value)`**, но не имеет свойств пересчета, то физическое значение из свойства **`5002 (Raw Value)`** записывается в инженерное значение (свойство **`2 (Value)`**).

8.1.2. Линейный пересчет

Линейный пересчет физического значения в инженерное и в обратном направлении значений выполняется по линейной зависимости. На рисунке ниже показан график линейной функции, отражающий зависимость физических значений параметров от инженерных значений и наоборот.

Линейная функция имеет вид:

```
y=kx+b
```



Для создания функций линейного пересчета при старте SePlatform.Data Server добавьте в конфигурацию сигнала следующие свойства:

- 5100 (нижняя граница физического значения);
- 5102 (верхняя граница физического значения);
- 5103 (нижняя граница инженерного значения);
- 5105 (верхняя граница инженерного значения).

При отсутствии указанных свойств значение сигнала не пересчитывается.

Линейный пересчет значений выполняется согласно следующей последовательности действий:

1. Вычисляются угловой коэффициент и начальная ордината прямой. Линейная функция имеет вид:

```
y=kx+b
```

где:

- k - угловой коэффициент;
- b - начальная ордината прямой.

Искомые величины вычисляются с использованием формулы:

$$\frac{x-x_1}{x_2-x_1} = \frac{y-y_1}{y_2-y_1}$$

где:

- x - физическое значение параметра;
- y - инженерное значение параметра;
- x_1, x_2 - координаты первой точки;
- y_1, y_2 - координаты второй точки.

Координатами точек по оси ординат являются значения нижней и верхней границы инженерного значения параметра (свойства **5103**, **5105**). Координатами точек по оси абсцисс являются значения нижней и верхней границы физического значения параметра (свойства **5100**, **5102**).

С использованием найденных величин будет сформировано уравнение прямой.

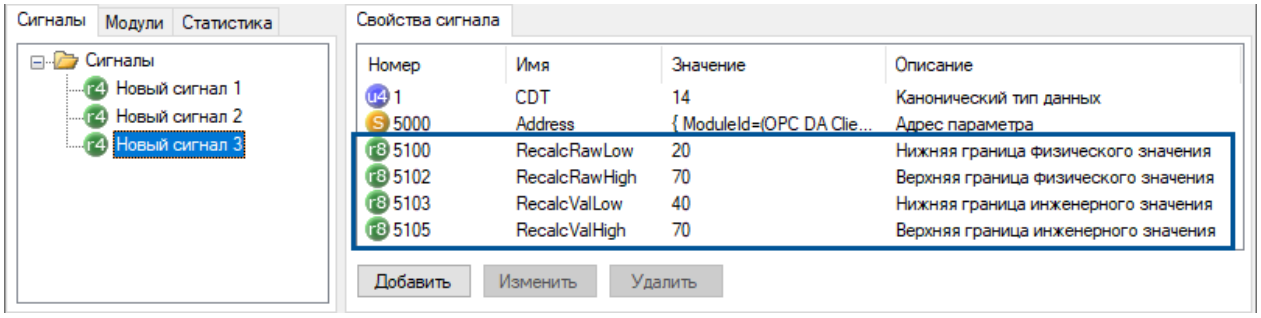
2. Согласно сформированному уравнению прямой строится график линейной функции в декартовой системе координат.

3. На основе построенного графика выполняется преобразование физического значения параметра в инженерное и наоборот.

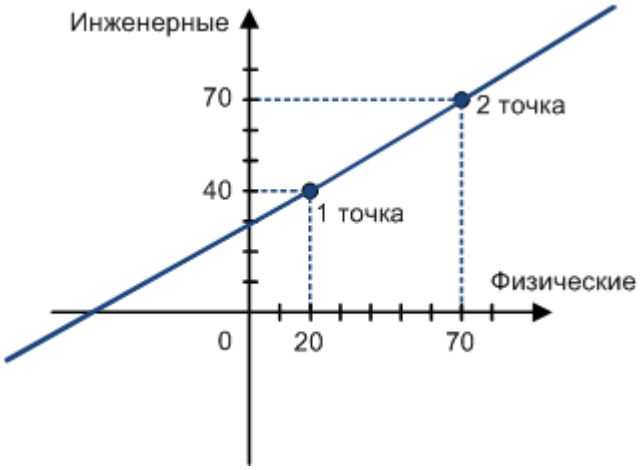
ПРИМЕР

Линейный пересчет значения сигнала

В конфигурацию сигнала добавлены свойства для линейного пересчета значения сигнала.



На рисунке ниже показан график с точками, которые являются границами пересчета. Первая точка имеет координаты, равные нижним границам физического и инженерного значений. Вторая точка имеет координаты, равные верхним границам физического и инженерного значений.



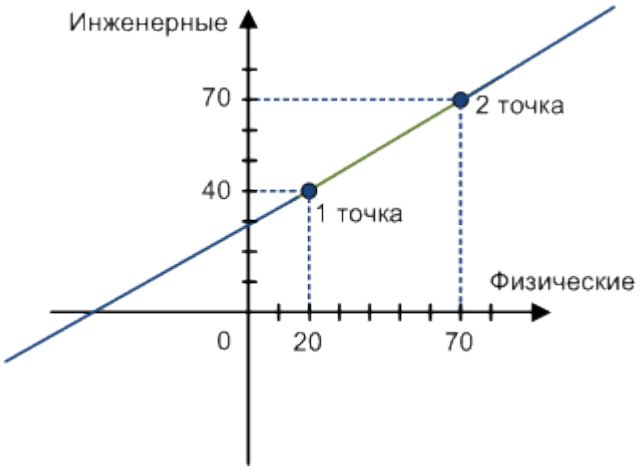
Из графика видно, что если физическое значение сигнала равно **«30»**, то после пересчета инженерное значение сигнала станет равно **«46»**.

Если физическое значение сигнала выходит за пределы границ пересчета, то пересчет значения сигнала продолжает осуществляться по линейной зависимости (например, физическое значение сигнала **«80»** будет пересчитано в инженерное значение сигнала **«76»**). Чтобы исключить пересчет значений сигналов за пределами границы пересчета, используйте свойство **`5106 (RecalcTruncate)`**.

Чтобы настроить линейный пересчет значений сигналов с обрезкой значений, добавьте в конфигурацию сигнала свойства **`5106 (RecalcTruncate)`** и **`5107 (RecalcSetFailureQuality)`**.

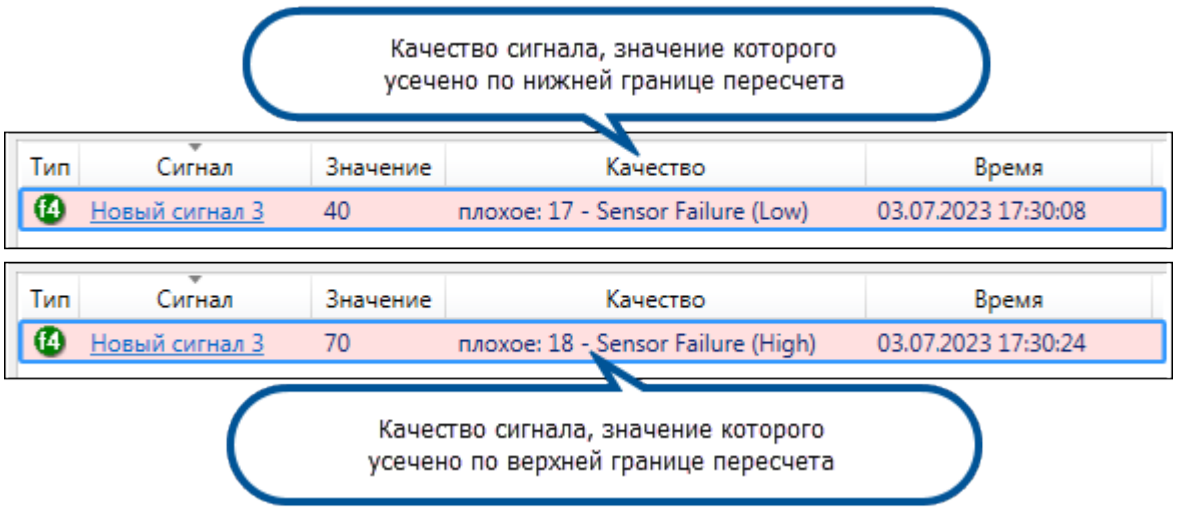
Чтобы значение, которое получится в результате пересчета, проверялось на принадлежность заданному диапазону пересчета, добавьте сигналу свойство **`5106 (RecalcTruncate)`**.

На рисунке ниже показан график из примера линейного пересчета сигнала, но с выделенным участком прямой, за рамки которого не должно выходить результирующее значение при пересчете. Участок прямой выделен зеленым цветом.



Если в результате пересчета полученное значение сигнала со свойством **5106 (RecalcTruncate)** вышло за пределы установленных границ, оно приравнивается к ближайшей границе пересчета. Качество сигнала устанавливается LIMIT_LOW, если значение было усечено до нижней границы, или LIMIT_HIGH, если значение было усечено до верхней границы.
Значение качества увеличивается на **«1»** в случае усечения по нижней границе и на **«2»** - по верхней. Если полученное значение находится в пределах границ пересчета, то качество сигнала не изменяется. На рисунке ниже показаны значения качества сигнала **«Новый сигнал 3»** (значения в сигнал поступают от DA-клиента).

Чтобы усеченным значеням сигналов выставлялось плохое качество, добавьте сигналу свойство **5107 (RecalcSetFailureQuality)**.
Если значение сигнала было усечено до нижней границы, сигналу выставляется качество Sensor Failure (Low).
Если значение было усечено до верхней границы, сигналу устанавливается качество Sensor Failure (High).
Усеченным значениям выставляется плохое значение качества, только если в сигнал добавлено свойство **5106 (RecalcTruncate)**.
На рисунке ниже показаны значения качества сигнала **«Новый сигнал 3»**.



8.1.3. Линейный пересчет с изломом

Линейный пересчет с изломом физического значения в инженерное и в обратном направлении значений выполняется по двум линейным зависимостям: до точки излома и после точки излома. Появление излома в прямой при пересчете объясняется наличием у некоторых приборов измерений двух диапазонов оцифровки.

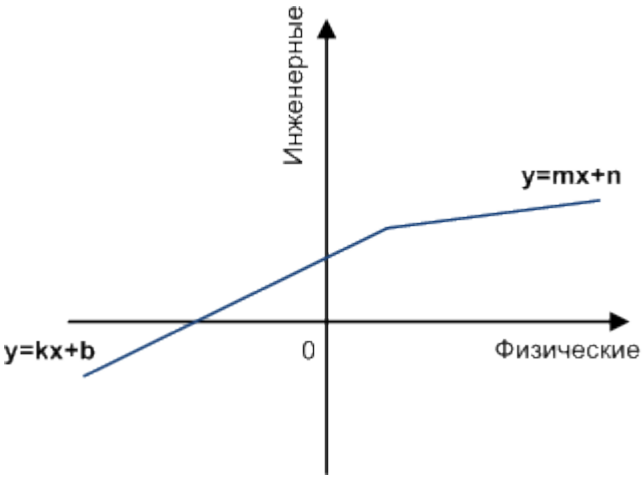
На рисунке ниже показан график, построенный с учетом двух линейных зависимостей. Аналогично линейному пересчету график отражает зависимость физических значений параметров от инженерных значений и наоборот.

Линейная функция на одном участке прямой имеет вид:

```
y=kx+b
```

На другом участке прямой вид:

```
y=mx+n
```



Для создания функций линейного пересчета с изломом при старте SePlatform.Data Server, добавьте в конфигурацию сигнала следующие свойства:

- **5100** (нижняя граница физического значения);
- **5101** (граница излома физического значения);
- **5102** (верхняя граница физического значения);
- **5103** (нижняя граница инженерного значения);
- **5104** (граница излома инженерного значения);
- **5105** (верхняя граница инженерного значения).

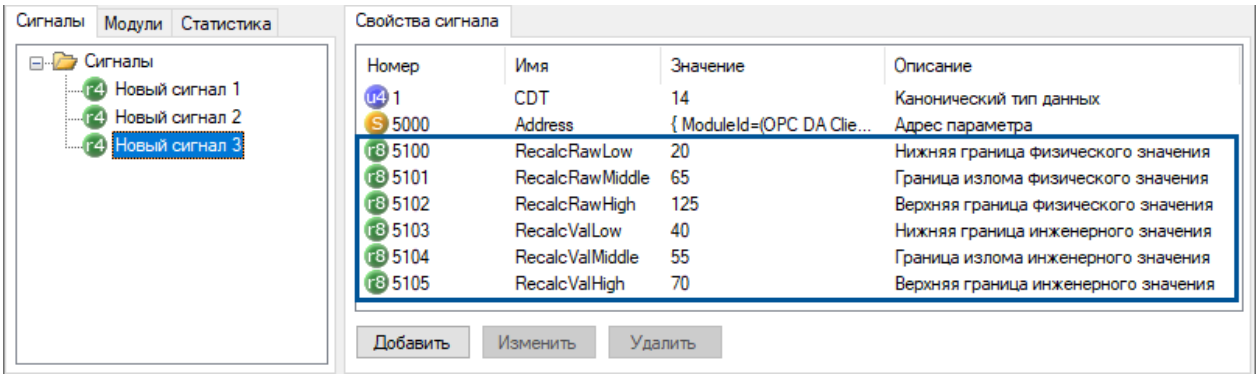
При отсутствии указанных свойств значение сигнала не пересчитывается.

Линейный пересчет с изломом представляет собой линейный пересчет на каждом участке прямой.

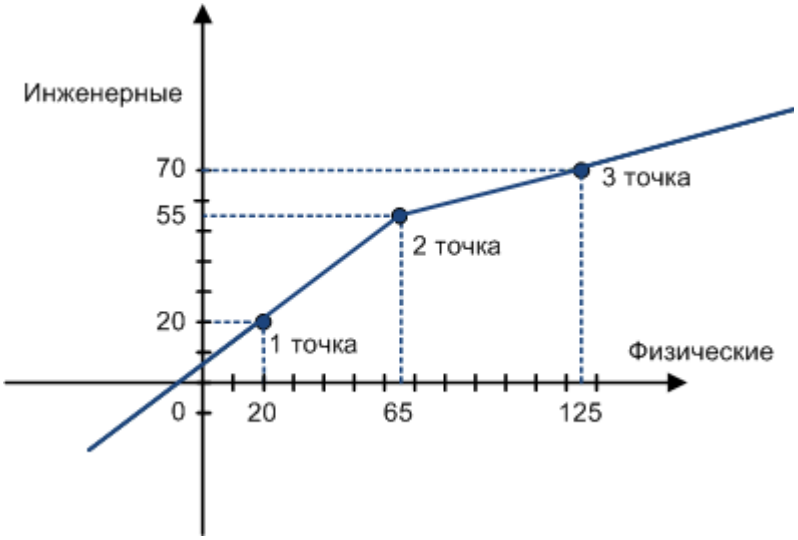
ПРИМЕР

Линейный пересчет значения сигнала с изломом

В конфигурацию сигнала добавлены свойства для линейного пересчета значения сигнала.



На рисунке ниже показан график, вторая точка которого является точкой излома, первая и третья точки являются границами пересчета.

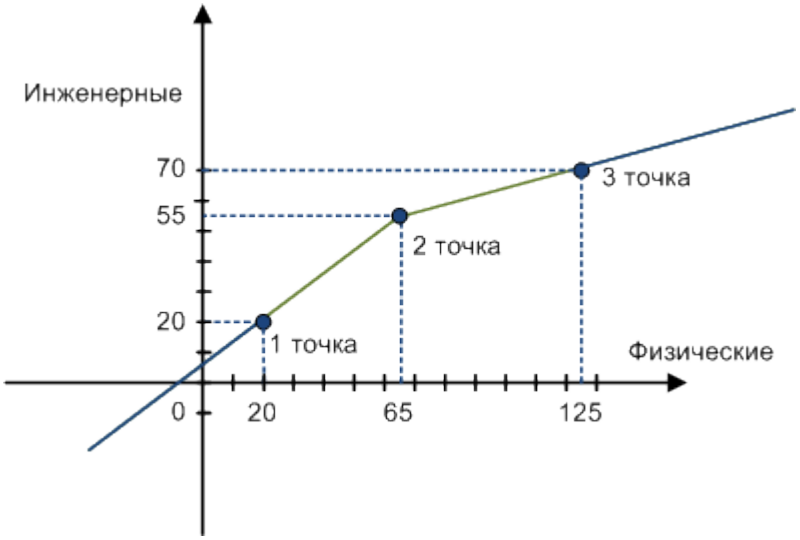


Из графика видно, что если физическое значение сигнала равно **«85»**, то после пересчета инженерное значение сигнала станет равно **«60»**.

Если физическое значение сигнала выходит за пределы границ пересчета, то пересчет значения сигнала продолжает осуществляться по линейной зависимости (например, физическое значение сигнала **«133»** будет пересчитано в инженерное значение сигнала **«72»**). Чтобы исключить пересчет значений сигналов за пределами границы пересчета, используйте свойство **5106 (RecalcTruncate)**.

Чтобы настроить линейный пересчет значений сигналов с изломом и с обрезкой значений, добавьте в конфигурацию сигнала свойства **5106 (RecalcTruncate)** и **5107 (RecalcSetFailureQuality)**. Свойства работают также, как в линейном пересчете значений сигналов.

На рисунке ниже показан график из примера линейного пересчета значения сигнала с изломом, но с выделенным участком кривой, за рамки которой не должно выходить результирующее значение при пересчете. Участок кривой выделен зеленым цветом.



8.1.4. Инверсия битовых сигналов

Действует только для сигналов, имеющих канонический тип bool (дискретный сигнал). Если сигнал является логическим, свойства пересчета не рассматриваются, при старте SePlatform.Data Server создается функция инверсного пересчета. Функция создается, если в конфигурации сигнала присутствует свойство **5108 (RecalcInvert)**.

При наличии у сигнала свойства **5108 (RecalcInvert)** исходное физическое значение сигнала, записанное в свойстве **5002 (Raw Value)**, инвертируется. Если значение сигнала было равно **«true»**, оно преобразуется в **«false»** и наоборот. Преобразованное значение записывается в свойство **2 (Value)**. Аналогичное правило действует для преобразования инженерных значений сигналов.

При отсутствии свойства **5108 (RecalcInvert)** у сигнала значение не инвертируется и записывается в свойство **5002 (Raw Value)**/ **2 (Value)** в исходном виде.

8.2. Перекладка значений сигналов

Для сигналов, поступающих с подчинённых станций (сигналы-источники), можно настроить перекладку данных в другие сигналы SePlatform.Data Server (сигналы-приёмники).

Перекладка - это один из видов вычислений, выполняемых ядром SePlatform.Data Server. Перекладка позволяет копировать:

- значение сигнала (**Value**);
- качество сигнала (**Quality**);
- метка времени сигнала (**Timestamp**);
- биты сигнала (часть значения сигнала).

Допускается перекладывать в один сигнал данные с разных сигналов (комбинированная перекладка).

Чтобы настроить перекладку, добавьте сигналу-приёмнику сигнальное свойство **6500** (тип string).

8.2.1. Перекладка значения

Перекладка значения сигнала позволяет скопировать значение свойства **Value (2)** сигнала-источника в сигнал-приёмник.

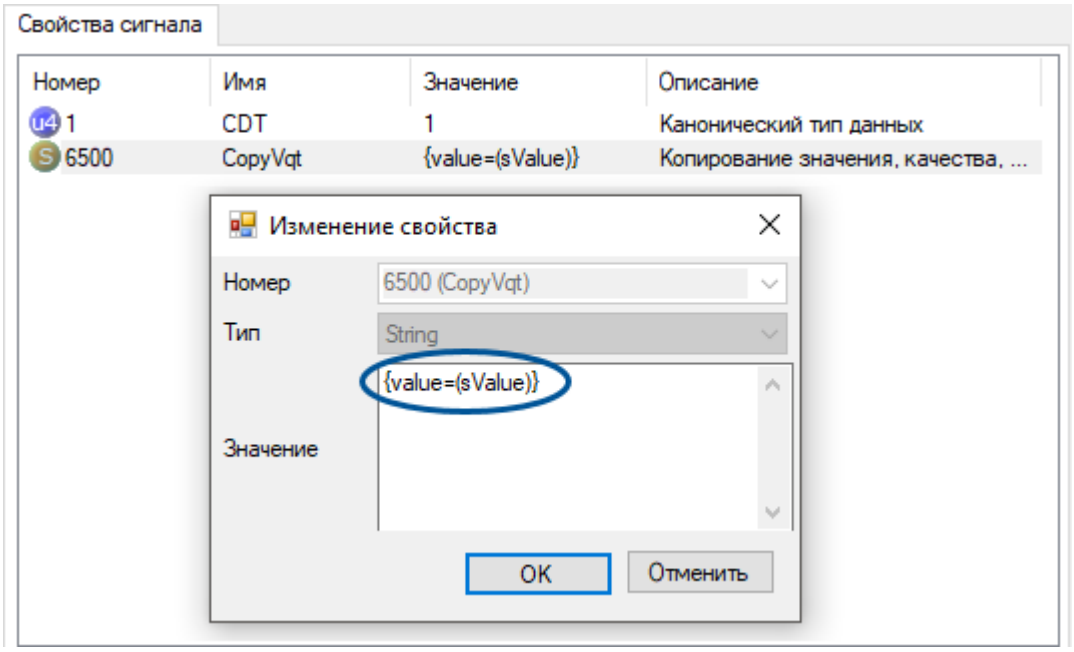
> ОБРАТИТЕ ВНИМАНИЕ
> Тип сигнала-приёмника должен совпадать с типом сигнала-источника.

Перекладка значений доступна для сигналов всех типов. Перекладка в сигнал-приёмник выполняется синхронно с изменением сигнала-источника.

Чтобы настроить перекладку значения:

1. Добавьте сигналу-приёмнику свойство **6500**.
2. В значении свойства укажите привязку к сигналу-источнику:

```
{value=(тег_сигнала-источника)}
```



Результаты перекладки:

- качество сигнала-приёмника:
  - GOOD (192), если значение сигнала-источника достоверно;
  - UNCERTAIN:SUB_NORMAL (88), если значение сигнала-источника недостоверно;
- метка времени сигнала-приёмника устанавливается равной метке времени сигнала-источника.

8.2.2. Полная перекладка

Полная перекладка позволяет скопировать свойства **Value (2)**, **Quality (3)** и **Timestamp (4)** сигнала-источника в сигнал-приёмник.

Полная перекладка может потребоваться, когда, например, необходимо VQT-значения, пришедшее с коммуникационного модуля, записать в несколько сигналов. В этом случае настраиваются один сигнал-источник на получение данных от модуля и несколько сигналов-приёмников для дублирования записи данных в них.

ОБРАТИТЕ ВНИМАНИЕ

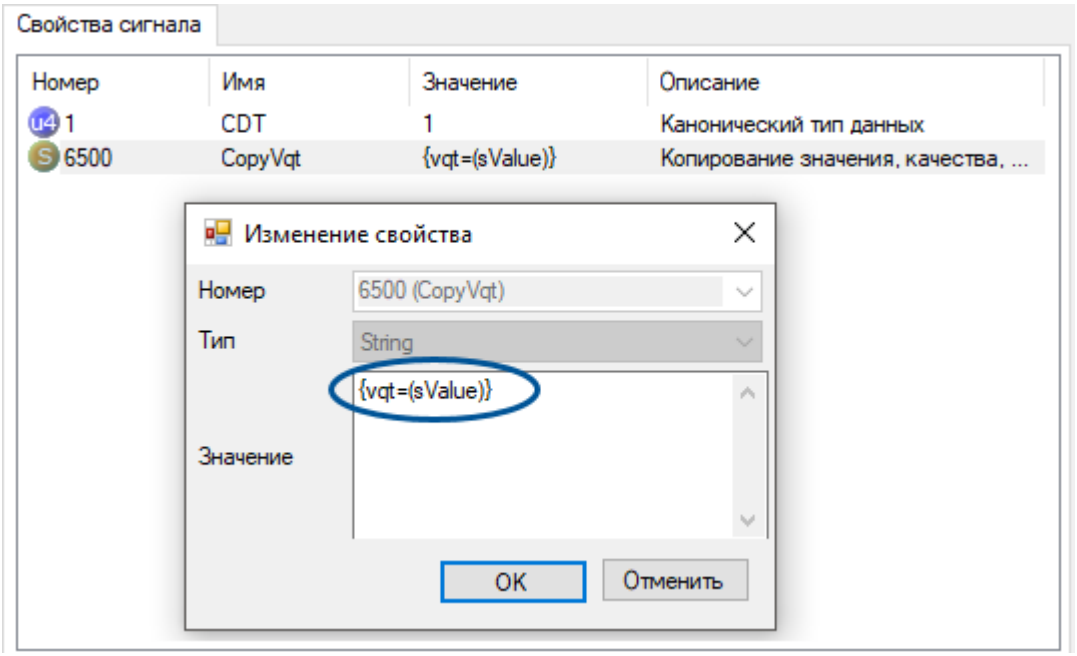
Тип сигнала-приёмника должен совпадать с типом сигнала-источника.

Перекладка значений доступна для сигналов всех типов. При изменении значения одного из VQT-свойств сигнала-источника изменяется значение соответствующего свойства сигнала-приёмника.

Чтобы настроить перекладку значения:

1. Добавьте сигналу-приёмнику свойство **6500**.
2. В значении свойства укажите привязку к сигналу-источнику:

```
{vqt=(тег_сигнала-источника)}
```



Результаты перекладки:

- значение (**Value**) сигнала-приёмника устанавливается равным значению сигнала-источника;
- качество (**Quality**) сигнала-приёмника устанавливается равным качеству сигнала-источника;
- метка времени (**Timestamp**) сигнала-приёмника устанавливается равной метке времени сигнала-источника.

8.2.3. Перекладка качества

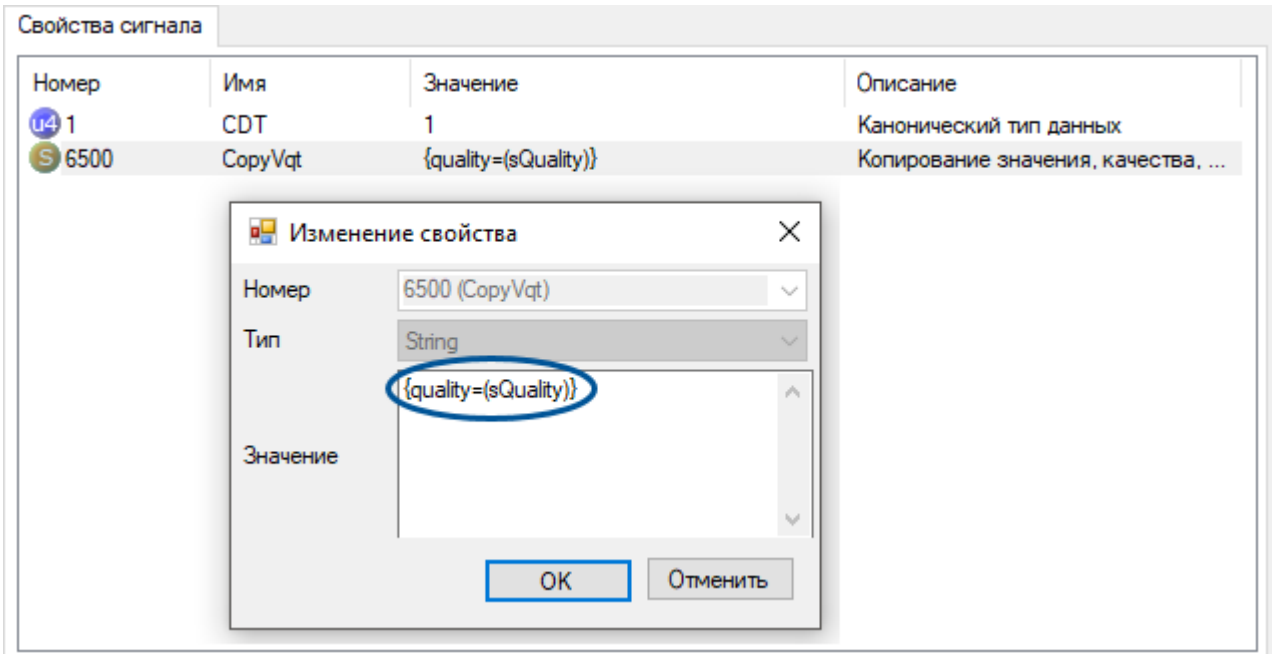
Перекладка качества позволяет устанавливать сигналу-приёмнику значение свойства **Quality (3)** посредством изменения значения сигнала-источника (**«true»**/**«false»**).

Тип сигнала-источника должен быть bool. Тип сигнала-приёмника может быть любой.

Чтобы настроить перекладку значения:

1. Добавьте сигналу-приёмнику свойство **6500**.
2. В значении свойства укажите привязку к сигналу-источнику:

```
{quality=(тег_сигнала-источника)}
```



Результаты перекладки:

- значение (**Value**) сигнала-приёмника остается неизменным;
- качество (**Quality**) сигнала приемника может быть следующим:
  - остается неизменным, если сигнал-приёмник имеет пустое значение (например, после запуска SePlatform.Data Server);
  - UNCERTAIN:SUB_NORMAL (88), если значение сигнала-источника недостоверно;
  - GOOD (192), если значение сигнала-источника достоверно и равно **«true»**;
  - BAD (0), если значение сигнала-источника достоверно и равно **«false»**.
- метка времени (**Timestamp**) сигнала-приёмника остается неизменной.

8.2.4. Перекладка битов

Перекладка бита или группы битов позволяет получить значение (**Value**) нужных битов из значения сигнала-источника.

Перекладка бита (группы битов) может потребоваться, когда, например, в сигнале-источнике хранится не собственное значение, а наборы битов (битовая маска) и необходимо получить значение одного конкретного бита (группы битов) из маски, который будет записан в сигнал-приёмник. На рисунке ниже показано, как значение третьего бита сигнала-источника передалось в сигнал-приёмник.

ОБРАТИТЕ ВНИМАНИЕ

При перекладке одного бита: тип сигнала-приёмника может быть bool или любым целочисленным типом. Тип сигнала-источника может быть любым, кроме string.

ОБРАТИТЕ ВНИМАНИЕ

При перекладке группы битов: типы сигнала-приёмника и сигнала-источника могут быть любого целочисленного типа. Тип сигнала-приёмника и сигнала-источника должны быть одного знака.

Перекладка в сигнал-приёмник выполняется синхронно с изменением сигнала-источника.

ОБРАТИТЕ ВНИМАНИЕ

Значения группы битов записываются в сигнал-приёмник в десятичном виде.

Чтобы настроить перекладку битов:

1. Добавьте сигналу-приёмнику свойство **6500**.
2. В значении свойства укажите привязку к сигналу-источнику (параметр **vqt** - для полной перекладки, параметр **value** - для перекладки только значений битов):

```
{vqt=(тег_сигнала-источника) bit=(номер_бита) count=(количество_битов)}
```

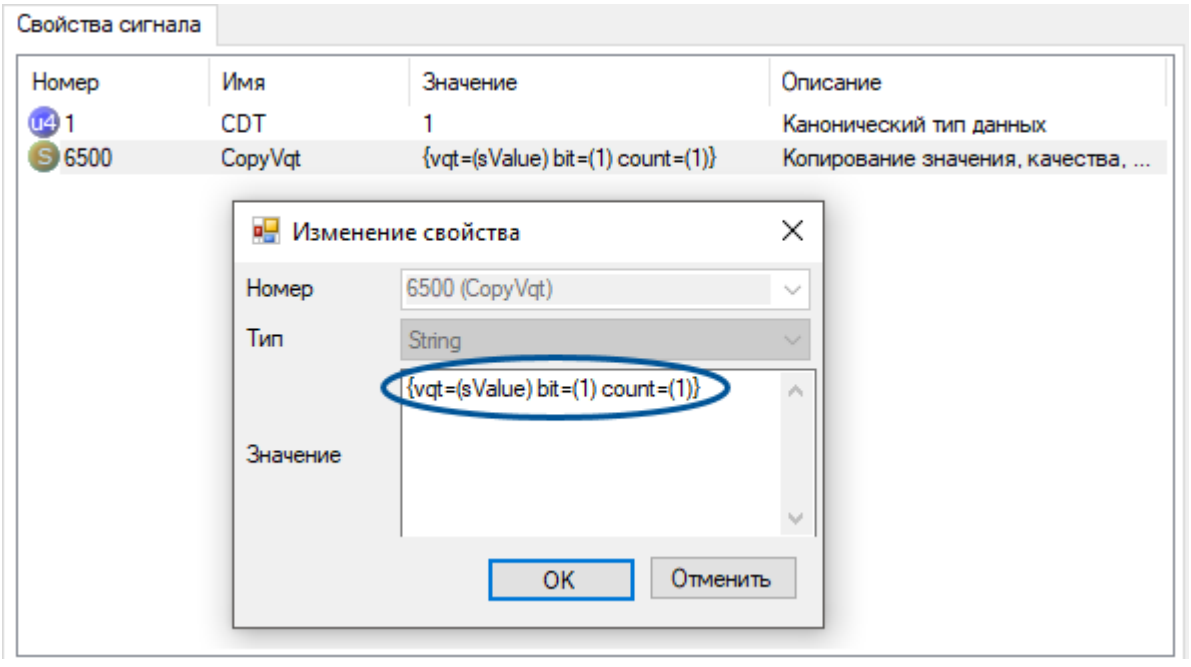
```
{value=(тег_сигнала-источника) bit=(номер_бита) count=(количество_битов)}
```

Где **bit** - номер бита сигнала-источника, с которого будет начинаться отсчет количества битов, указанных в параметре **count**.

ОБРАТИТЕ ВНИМАНИЕ

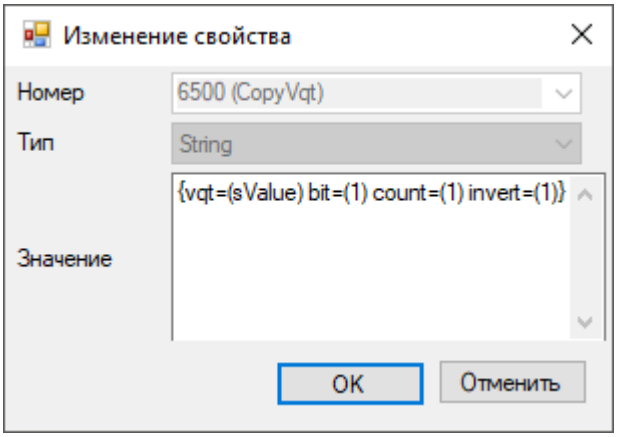
Нумерация битов ведется с **«0»**, где **«0»** соответствует младшему биту.

На рисунке ниже показана настройка перекладки VQT-значений только второго бита сигнала **«sValue»** в сигнал-приёмник.



Если при перекладке одного бита необходимо инвертировать полученное значение перед записью в сигнал-приёмник, то используйте параметр **invert** (возможные значения параметра: **«1»** (true) или **«0»** (false)).

```
{vqt=(sValue) bit=(1) count=(1) invert=(1)}
```



В результате перекладки:

- значение сигнала-приёмника устанавливается равным значению указанного бита в значении сигнала-источника;
- качество сигнала-приёмника:
  - GOOD (192), если значение сигнала-источника достоверно;
  - UNCERTAIN:SUB_NORMAL (88), если значение сигнала-источника недостоверно;
  - качество сигнала-приёмника устанавливается равным качеству сигнала-источника, если настроена полная перекладка битов;
- метка времени сигнала-приёмника устанавливается равной метке времени сигнала-источника.

Перекладка битов не выполняется и в журнал сообщений выдаются соответствующие сообщения, если:

- число (**count** + **bit**) больше размера типа сигнала-источника;
- число **count** больше размера типа сигнала-приёмника;
- несовпадение наличия знака в типе сигнала-источника и сигнала-приёмника.

8.2.5. Комбинированная перекладка

Комбинированная перекладка позволяет получать данные с нескольких сигналов параллельно и записывать в один сигнал.

Комбинированная перекладка может потребоваться, когда, например, по коммуникационному протоколу значения **Value** и **Quality** поступают в разные сигналы. Чтобы получить единое значение для сигнала используется комбинированная перекладка (значение сигнала берётся из одного сигнала, а качество - из другого).

Возможны следующие варианты комбинаций:

- перекладка значения и установка качества. Значение свойства **6500**:

```
{value=(тег_сигнала-источника)}{quality=(тег_сигнала-источника)}
```

- установка качества и перекладка указанного бита. Значение свойства **6500**:

```
{vqt=(тег_сигнала-источника) bit=(номер_бита) count=(количество_битов)}{quality=(тег_
сигнала-источника)}
```

Определение итогового качества сигнала-приёмника (при использовании варианта с перекладкой значения и установкой качества) выполняется по следующим правилам:

- по правилам перекладки значения определяется качество сигнала-приёмника;
- по правилам перекладки качества определяется качество сигнала-приёмника;
- если оба определенных качества достоверны, для сигнала-приёмника устанавливается качество GOOD (192);
- если оба или какой-то один из определенных качеств недостоверно, для сигнала-приёмника устанавливается наименее достоверное из них.

8.2.6. Относительная адресация при копировании

При работе с однотипными деревьями сигналов использовать абсолютную адресацию неудобно: при копировании/размножении ветки сигналов необходимо редактировать ссылки на сигналы-источники во всех скопированных сигналах-приёмниках.

Для быстрого конфигурирования однотипных веток сигналов используйте относительную адресацию, при которой ссылки на сигналы-источники определяются относительно сигнала-приёмника. Использование относительной адресации позволяет:

- копировать ветки сигналов без корректировки ссылок на сигналы-источники;
- изменять названия узлов дерева сигналов без нарушения логики работы функции копирования.

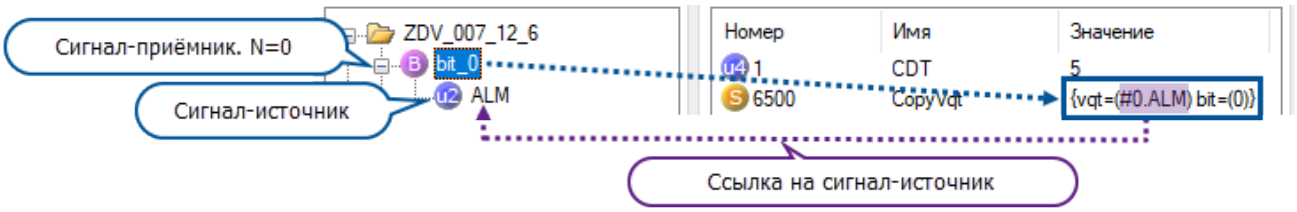
Чтобы настроить относительную адресацию при копировании, укажите в значении свойства **6500** сигнала-приёмника запись следующего формата:

```
#<N>.<имя_сигнала-источника>
```

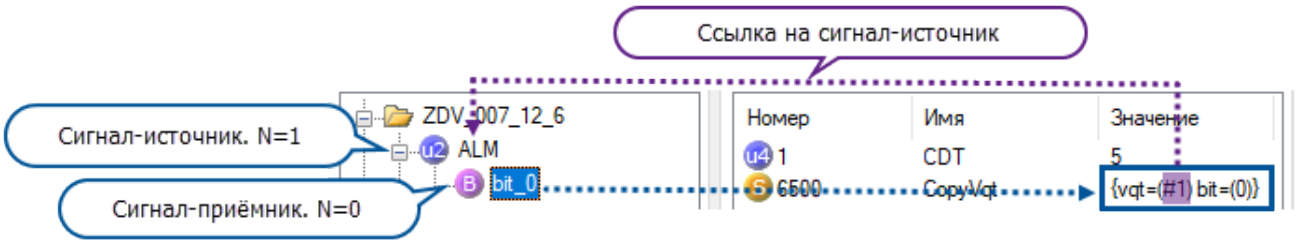
где **#<N>** - обращение к N-ому уровню дерева, на котором расположен сигнал-источник. Отсчет уровней ведётся от сигнала-приёмника, уровень которого является нулевым. Количество уровней неограниченно (от 0 до ∞).

Примеры обращений к сигналу-источнику:

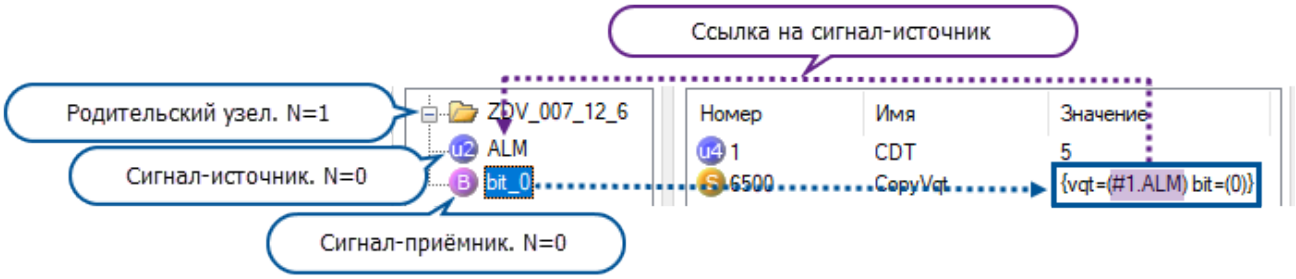
- **#0.<имя_сигнала-источника>** - обращение к сигналу-источнику, расположенному на уровень ниже сигнала-приёмника;



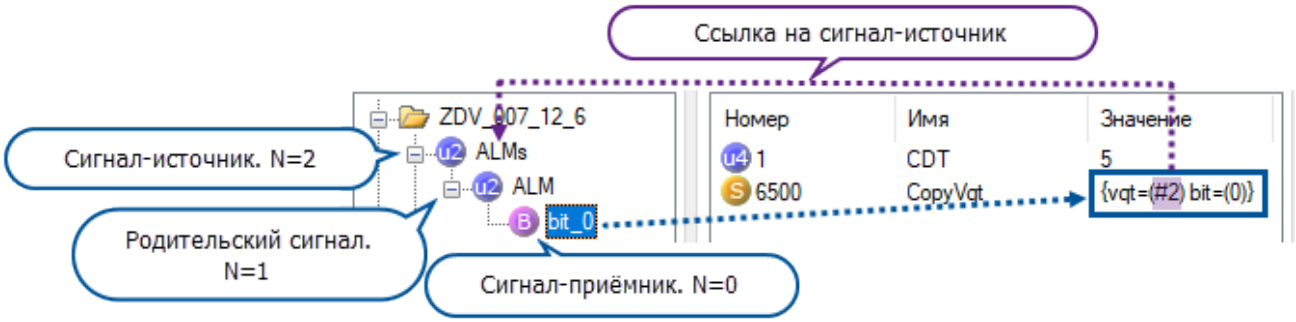
- **#1** - обращение к сигналу-источнику, расположенному на уровень выше сигнала-приёмника;



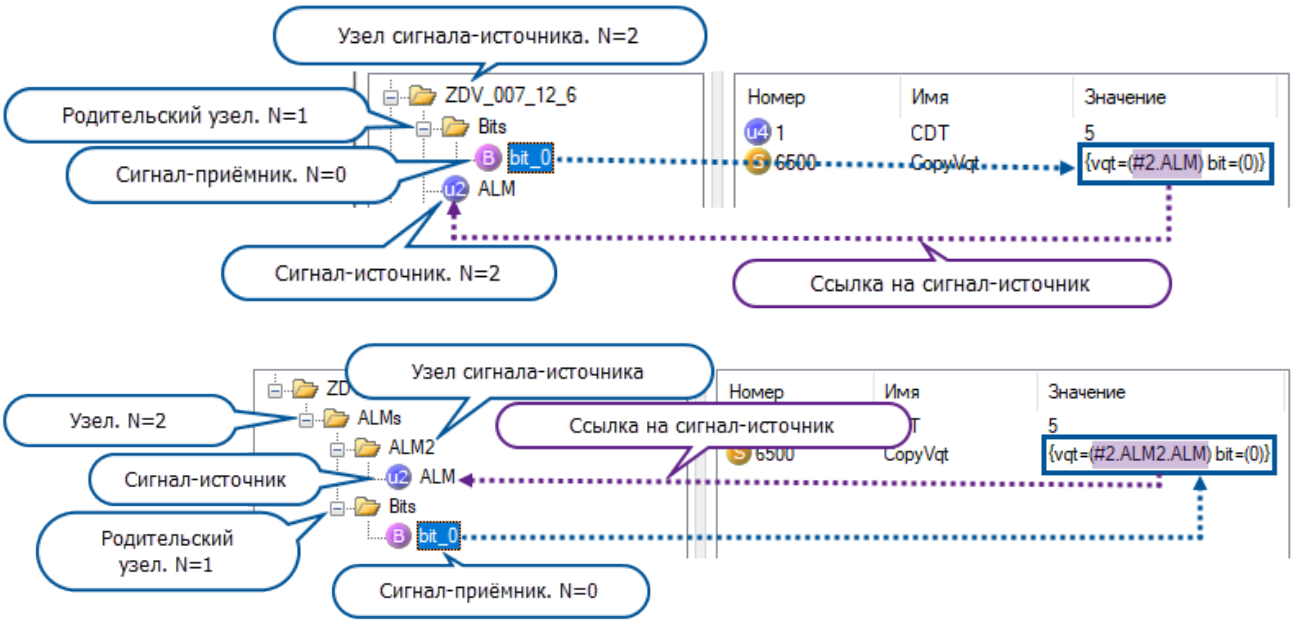
- **#1.<имя_сигнала-источника>** - обращение к сигналу-источнику, расположенному в одном узле с сигналом-приёмником;



- **#2** - обращение к сигналу-источнику, расположенному на два уровня выше сигнала-приёмника;



- **#2.<имя_сигнала-источника>** - обращение к сигналу-источнику, расположенному в узле на два уровня выше сигнала-приёмника:



Использовать относительную адресацию можно для всех видов копирования значений сигнала.

Пример использования относительной адресации

В конфигурацию сервера необходимо добавить сигналы для приёма уведомлений о работе для четырёх однотипных задвижек:

- ZDV_007_12_3;
- ZDV_007_12_4;
- ZDV_007_12_5;
- ZDV_007_12_6.

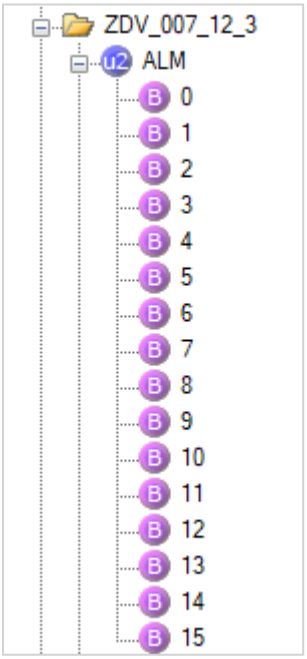
Для каждой задвижки необходимо добавить сигналы типа uint2, принимающие уведомления об авариях, режимах и состояниях:

- **«ALM»**;
- **«Mode»**;
- **«State»**.

Для каждого сигнала необходимо добавить дочерние сигналы (**«0»** ... **«15»**) типа bool, в которые будут копироваться значения соответствующих битов родительского сигнала. Родительские сигналы **«ALM»**, **«Mode»**, **«State»** являются сигналами-источниками для дочерних сигналов-приёмников **«0»** ... **«15»**.

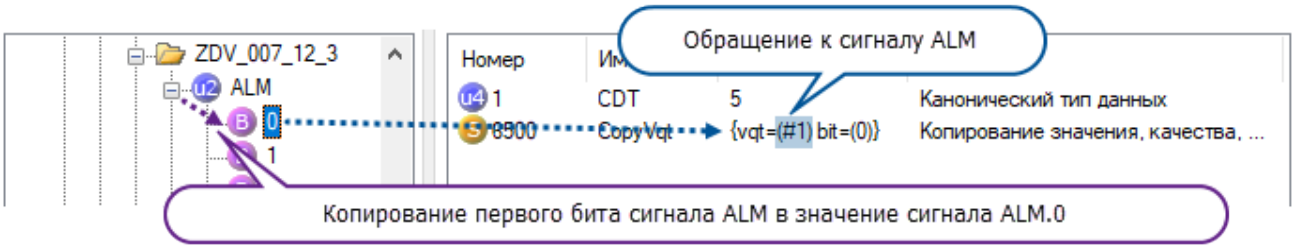
Порядок работы:

1. Создайте узел сигналов (папку) для задвижки ZDV_007_12_3.
2. В узле задвижки ZDV_007_12_3 создайте сигнал-источник **«ALM»** и задайте для данного сигнала параметры адреса в свойстве **5000 (Address)**.
3. Добавьте для сигнала **«ALM»** дочерние сигналы-приёмники **«0»** ... **«15»**. Дерево сигналов задвижки ZDV_007_12_3 имеет вид:

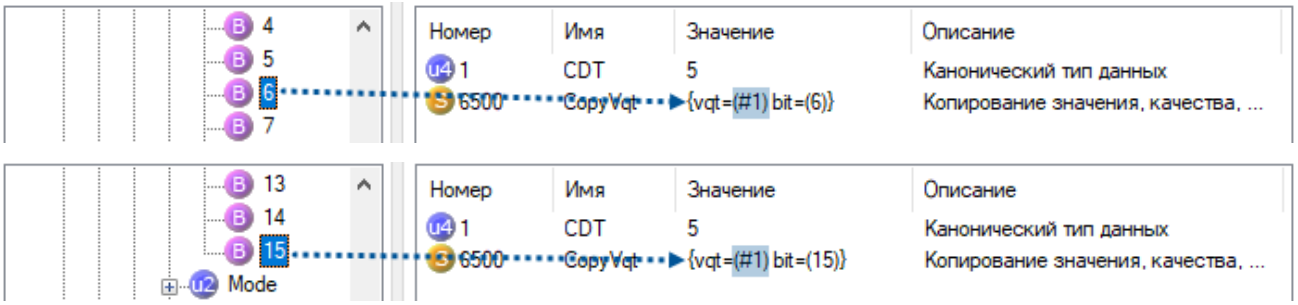


4. В значении свойства **6500 (Copy Vqt)** для каждого сигнала-приёмника укажите параметры копирования соответствующего бита сигнала-источника с использованием относительной адресации. Например, для сигнала **«ALM.0»** свойство **6500 (Copy Vqt)** имеет следующее значение:

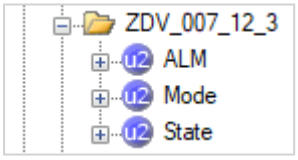
```
{vqt=(#1) bit=(0)}
```



На рисунках ниже приведены примеры настройки сигналов-приёмников **«ALM.6»** и **«ALM.15»**.



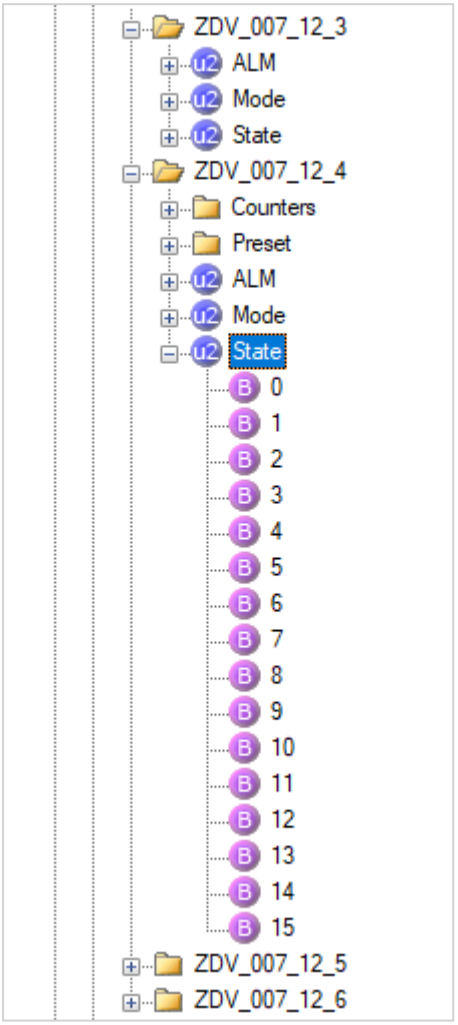
5. После настройки копирования битов сигнала **«ALM»** с использованием относительной адресации создайте 2 копии сигнала **«ALM»** в узле задвижки ZDV_007_12_3 и измените имена сигналов на **«Mode»** и **«State»**.



Для скопированных сигналов-источников укажите соответствующие параметры адресов в свойстве **5000 (Address)**.

6. Создайте 3 копии узла сигналов задвижки ZDV_007_12_3 и задайте им соответствующие имена:
   - ZDV_007_12_4;
   - ZDV_007_12_5;
   - ZDV_007_12_6.

Итоговый вид дерева сигналов показан ниже:



После размножения узлов сигналов задвижек следует изменить только параметры адресов в свойстве **`5000 (Address)`** для сигналов **«ALM»**, **«Mode»** и **«State»** каждой задвижки. Никаких изменений в настройке копирования битов сигналов не требуется.

8.2.7. Пример перекладки

Требуется получить значение давления с КП. Значение давления записывается в сигнал **«pressure»** (тип UInt1). Тег сигнала **«AK.DMN.R_Bel.LU_21_24.KP_002.pressure»**.

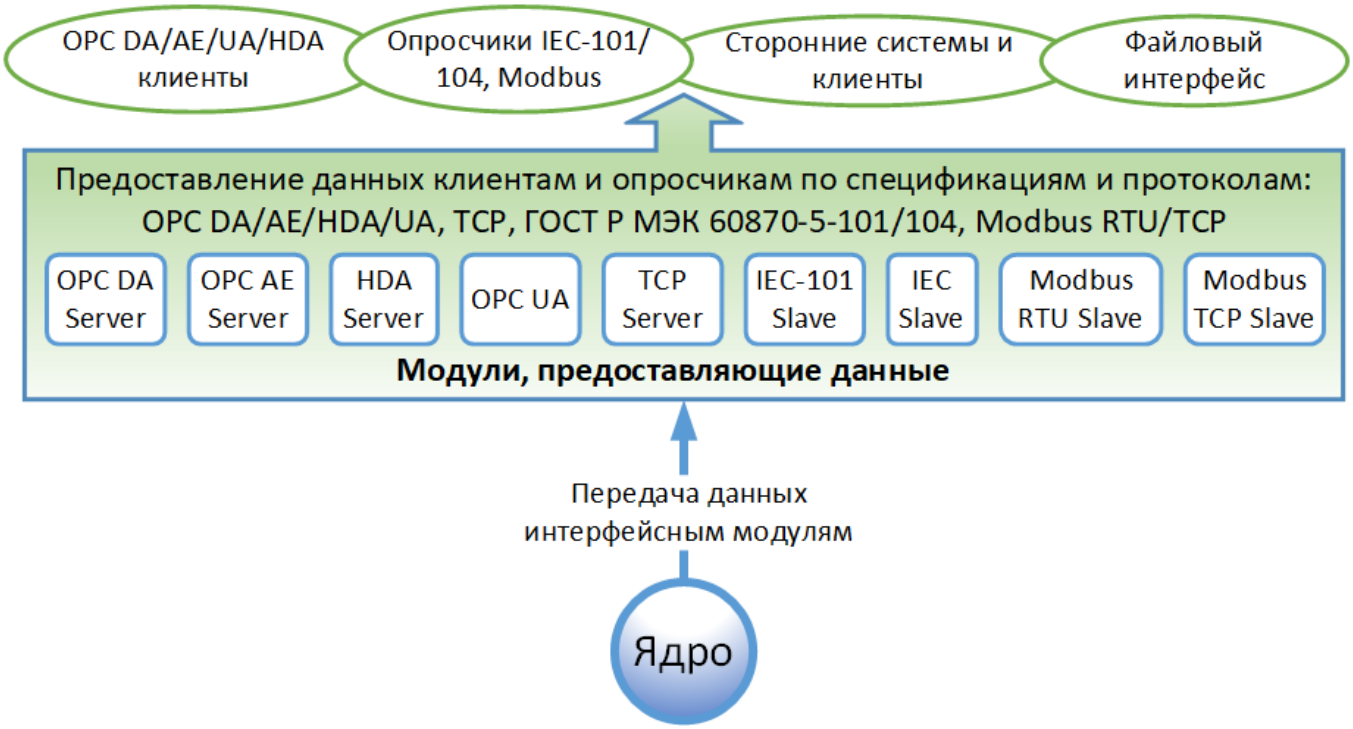
Чтобы настроить перекладку:

1. Создайте сигнал **«p_value»** (тип UInt1), в который будет записываться значение сигнала **«pressure»**.
2. Добавьте сигналу **«p_value»** свойство **6500**.
3. Задайте значение свойства **6500**:

```
{value=(AK.DMN.R_Bel.LU_21_24.KP_002.pressure)}
```

9. Предоставление данных

Предоставление обработанных данных клиентам, опросчикам и сторонним системам - это одна из основных задач SePlatform.Data Server. Схема передачи данных из ядра SePlatform.Data Server по запросам клиентов показана ниже.



SePlatform.Data Server способен предоставлять данные по различным коммуникационным протоколам и спецификациям.

Чтобы настроить SePlatform.Data Server на предоставление данных:

1. Добавьте в состав конфигурации модули, которые соответствуют выбранным протоколам/спецификациям, и активируйте их.
2. Настройте модули на предоставление данных. Как настроить модули см. в документе на соответствующий модуль.

10. Приложения

Приложение A: Свойства сигналов SePlatform.Data Server

ID	Тип	Короткое имя	Описание
Стандартные OPC свойства			
1	uint4	CDT	CDT (Канонический тип данных)
2	variant	Value	Значение
3	uint4	Quality	Качество
4	STL::time	Timestamp	Метка времени
5	uint4	AccRight	Права доступа. Значения свойства: › 1 - readable - чтение; › 2 - writable - запись; › 3 - readWritable - чтение/запись. Если значение свойства не задано, то при старте сервера свойство создаётся динамически со значением **«readWritable»**
6	float	ScanRate	Скорость обновления (сканирования)
100	string	EUnit	Единицы измерения
101	string	Description	Описание сигнала
6500	string	CopyVqt	Записывать в сигнал перекладываемое значение
Коммуникационные модули (адресация сигнала)			
5000	string	Address	Адрес сигнала
5001	string	Active	Активный протокол
5002	variant	RawValue	Физическое значение. Свойство создается сервером динамически. Тип свойства должен соответствовать каноническому типу сигнала. При создании свойства активируются функции пересчета в инженерное значение (свойство **2 Value**) и обратно
Пересчет			
5100	double	RecalcRawLow	Нижняя граница физического значения
5101	double	RecalcRawMiddle	Граница излома физического значения
5102	double	RecalcRawHigh	Верхняя граница физического значения

ID	Тип	Короткое имя	Описание
5103	double	RecalcValLow	Нижняя граница инженерного значения
5104	double	RecalcValMiddle	Граница излома инженерного значения
5105	double	RecalcValHigh	Верхняя граница инженерного значения
5106	bool	RecalcTruncate	Усекать значение по границе пересчета и добавлять в качество флаги усечения (LIMIT_LOW или LIMIT_HIGH)
5107	bool	RecalcSetFailureQuality	При усечении по границе пересчета выставлять SENSOR_FAILURE
5108	bool	RecalcInvert	Инвертировать логическое значение. Действует только для сигналов с типом bool
Ссылки на объекты			
6001	string		Полное имя объекта, к которому ведёт данная ссылка
6002	uint4		Разновидность ссылки. Значения свойства: › **«0»** - ссылка ведёт исключительно на указанный объект и не затрагивает его поддерево сигналов; › **«1»** - ссылка ведёт на указанный объект и его поддерево сигналов.
6003	bool		Автораскрытие ссылки. Значения свойства: › **«true»** - пользователю предоставляется возможность в DA-клиенте раскрыть поддерево объекта, на который ведёт ссылка; › **«false»** - невозможность раскрыть в DA-клиенте поддерево объекта, на который ведёт ссылка.
6004	bool		Константность ссылки. Значения свойств: › **«true»** - сигналы объекта, на который ведёт ссылка, доступны только для чтения в DA-клиенте; › **«false»** - сигналы объекта, на который ведёт ссылка, доступны для изменений через DA-клиент.
6005	bool		Свойство модуля OPC AE Server. Если у объекта определено данное свойство (значение **«true»**), то агрегатор области, в которой определена ссылка, агрегирует также события целевого объекта
Восприимчивость сигнала к изменениям			

ID	Тип	Короткое имя	Описание
6100	string		Восприимчивость сигнала к изменениям. Значения свойства: › **«VQChange»** - значение сигнала считается изменившимся, если изменилось значение хотя бы одного из свойств сигнала: › значение **2 (Value)**; › качество **3 (Quality)**; › **«AnyChange»** - значение сигнала считается изменившимся, если изменилось значение хотя бы одного из свойств сигнала: › значение **2 (Value)**; › качество **3 (Quality)**; › метка времени **4 (Timestamp)**; › **«Repeat»** - значение сигнала считается изменившимся даже при полном повторе значений свойств сигнала: › значение **2 (Value)**; › качество **3 (Quality)**; › метка времени **4 (Timestamp)**.
Ведение детального журнала изменений сигналов			
7000	bool		Ведение детального журнала изменений сигналов
Резервирование			
8000	bool		Опциональная синхронизация при резервировании
Ведение истории			
9001	bool	Historizing	Ведение истории
9002	string	HistoryParams	Дополнительные параметры сохранения истории
Модуль Write VQT			
10000	bool	EnableWriteVqt	Постановка на обслуживание сигнала модулю Write VQT
Модуль OPC UA Client			
11000	uint1		Преобразование входящих значений типа ByteString в строку: › **«0»** - не принимать данные типа ByteString; › **«1»** - принимать данные типа ByteString и преобразовывать в строку.
Модуль логики			

ID	Тип	Короткое имя	Описание
777001-777003	string		Используются в конфигурациях, созданных в SePlatform.Development Studio с использованием объектной модели. Не предназначены для редактирования пользователем
777005	string		Содержит определения внешних функций
777006	string		Содержит карту дескрипторов сигнатур внешних функций
777010	string		Формула для вычисления значения сигнала
777011	string		Условие активации процедуры, определенной в свойстве 777012
777012	string		Код процедуры на языке SePlatform.Om. которая активируется по условию свойства **777011**
777013	string		Обработчик, срабатывающий перед отправкой события. Срабатывает в тот момент, когда по сигналу уже сгенерировано событие, но само событие ещё не отправлено клиентам
777015	uint4		Значение таймера (в миллисекундах) для исполнения процедуры, которая содержится в свойстве 777016
777016	string		Код процедуры на языке SePlatform.Om, исполняемый циклически по таймеру. Значение таймера указывается в свойстве **777015**
777017	string		Содержит cron-выражение для выполнения процедуры по расписанию. Код самой процедуры содержится в свойстве **777018**
777018	string		Содержит код процедуры, которая будет выполняться по расписанию, заданному в формате cron-выражения в свойстве **777017**
777020-777051	string		Используются в конфигурациях, созданных в SePlatform.Development Studio с использованием объектной модели. Не предназначены для редактирования пользователем
Системные (внутренние) свойства			
6000	uint4	NotAckEventCount	Количество неквитированных событий
999000	string	ObjectType	Тип объекта
999001	uint4	ObjectCode	Код объекта

ID	Тип	Короткое имя	Описание
999002	string	ObjectSound	Звук объекта
999003	bool	EventsEnabled	Признак генерации событий
999004	string	Conditions	Условия генерации событий
999005	bool	IsAbstract	Тип абстрактный или нет (Экземпляры абстрактного типа создавать нельзя)

Список терминов и сокращений

OPC (OLE for Process Control)	Программная технология на базе OLE, ActiveX, COM/DCOM, предоставляющая набор объектов, используемых в автоматизации технологических процессов, и интерфейсов доступа к ним.
OPC AE (OLE for Process Control Alarms&Events)	Интерфейс передачи сообщений OPC, предоставляет функции уведомления по требованию о различных событиях: аварийные ситуации, действия оператора, информационные сообщения и другие.
OPC DA (OLE for Process Control Data Access)	Интерфейс передачи сигналов OPC, описывает набор функций обмена данными в реальном времени.
OPC UA (OPC Unified Architecture)	Унифицированная спецификация, определяющая передачу данных в промышленных сетях.
OPC сервер	Сервер, предоставляющий доступ по интерфейсам OPC к технологическим данным, полученным по различным каналам, главным образом, от среднего уровня АСУ ТП. Обычно разрабатывается производителем контроллеров, автоматики.
TCP (Transmission Control Protocol)	Протокол управления передачей.
АСУ ТП	Автоматизированная система управления технологическими процессами.
АСУП	Автоматизированная система управления предприятием.
Качество	Характеристика достоверности сигнала.
КП	Контрольный, контролируемый пункт, клапан перепускной.
ОБД	Оперативная база данных параметров.
ОС	Операционная система.
Сигнал	Единица технологической информации, обладающая определённым набором обязательных и дополнительных свойств.